# Final Project

Roger Castillo

12/10/2023

```
country <- read.csv('sample-data.csv')
```

## Data Pre-processing

- Renaming long column names

```
names(country)
```

```
##  [1] "X.1"                    "X"                      "Country"
##  [4] "Date"                   "Annual.GDP"             "GDP.per.capita"
##  [7] "Debt"                   "Debt.Per.Capita"        "Deficit...M..."
## [10] "Expenditure..M..."      "Expenditure.Per.Capita" "Corruption.Index"
## [13] "Exports"                "Exports...GDP"          "Imports"
## [16] "Imports...GDP"          "Population"             "Fertility.Rate"
## [19] "Crude.death.rate"       "HDI"                    "Life.expectancy"
## [22] "CO2.Tons.per.capita"    "Continent"              "status"
```

```
names(country[,9:10])
```

```
## [1] "Deficit...M..."    "Expenditure..M..."
```

```
names(country)[names(country) == "Expenditure..M..."] <- "Expenditure"
names(country)[names(country) == "Deficit...M..."] <- "Deficit"
names(country[,9:10])
```

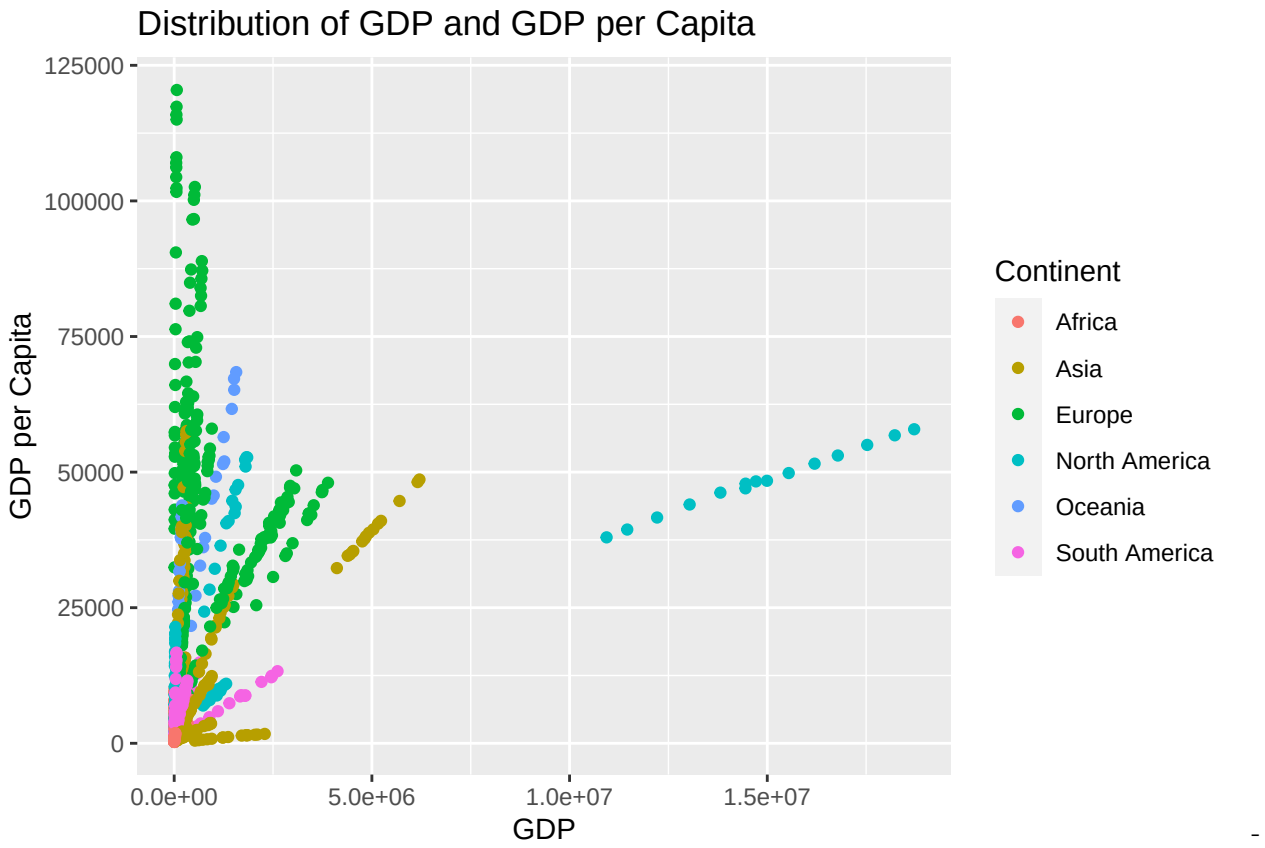
```
## [1] "Deficit"     "Expenditure"
```

```
country <- na.omit(country)
```

## Economic Analysis

- What is the distribution of GDP and GDP per capita across continents and countries?
- Is there a correlation between debt and GDP or debt per capita and GDP per capita?

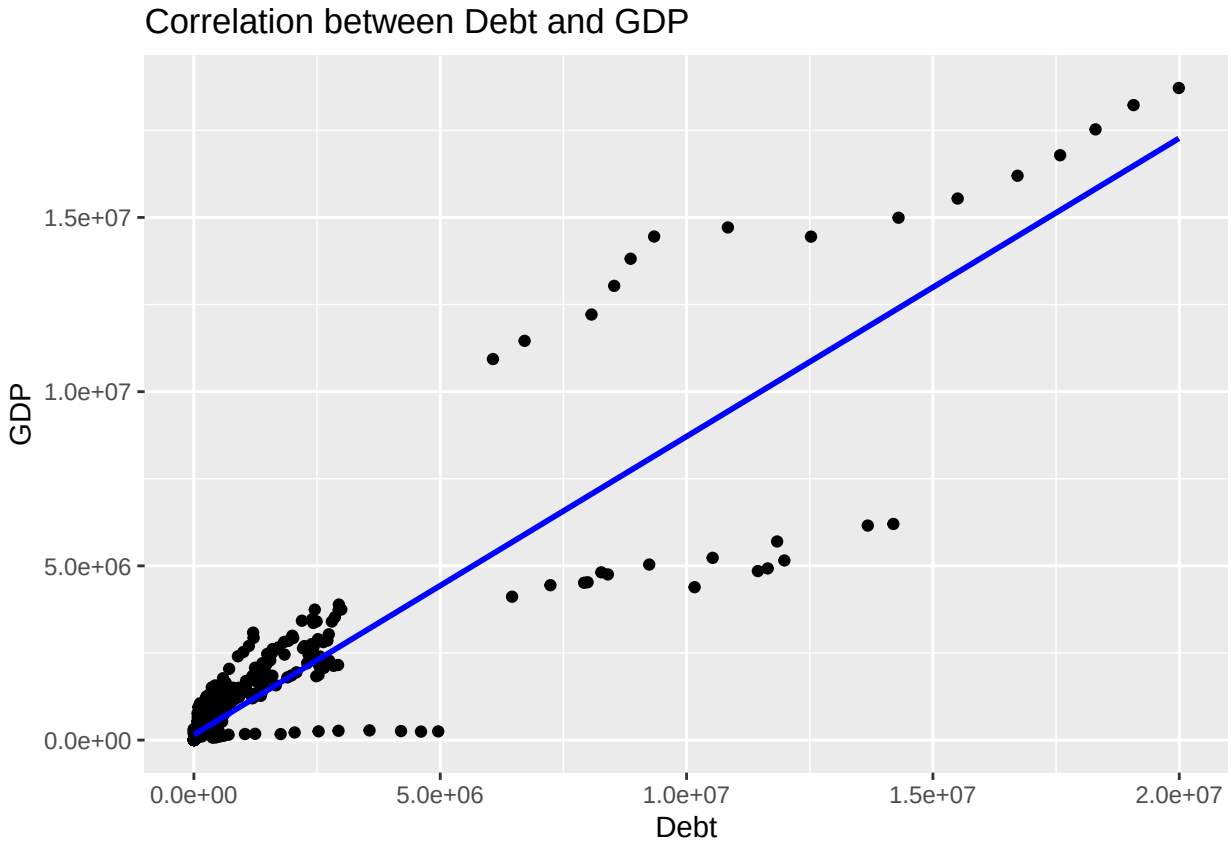
```
library(ggplot2)
# Scatter plot for GDP vs. GDP per capita
ggplot(country, aes(x = Annual.GDP, y = GDP.per.capita, color = Continent)) +
  geom_point() +
  labs(title = "Distribution of GDP and GDP per Capita",
       x = "GDP",
       y = "GDP per Capita",
       color = "Continent")
```




Higher levels of GDP are seen across north America. - Europe has the highest levels of of GDP per capita despite have a low annual GDP

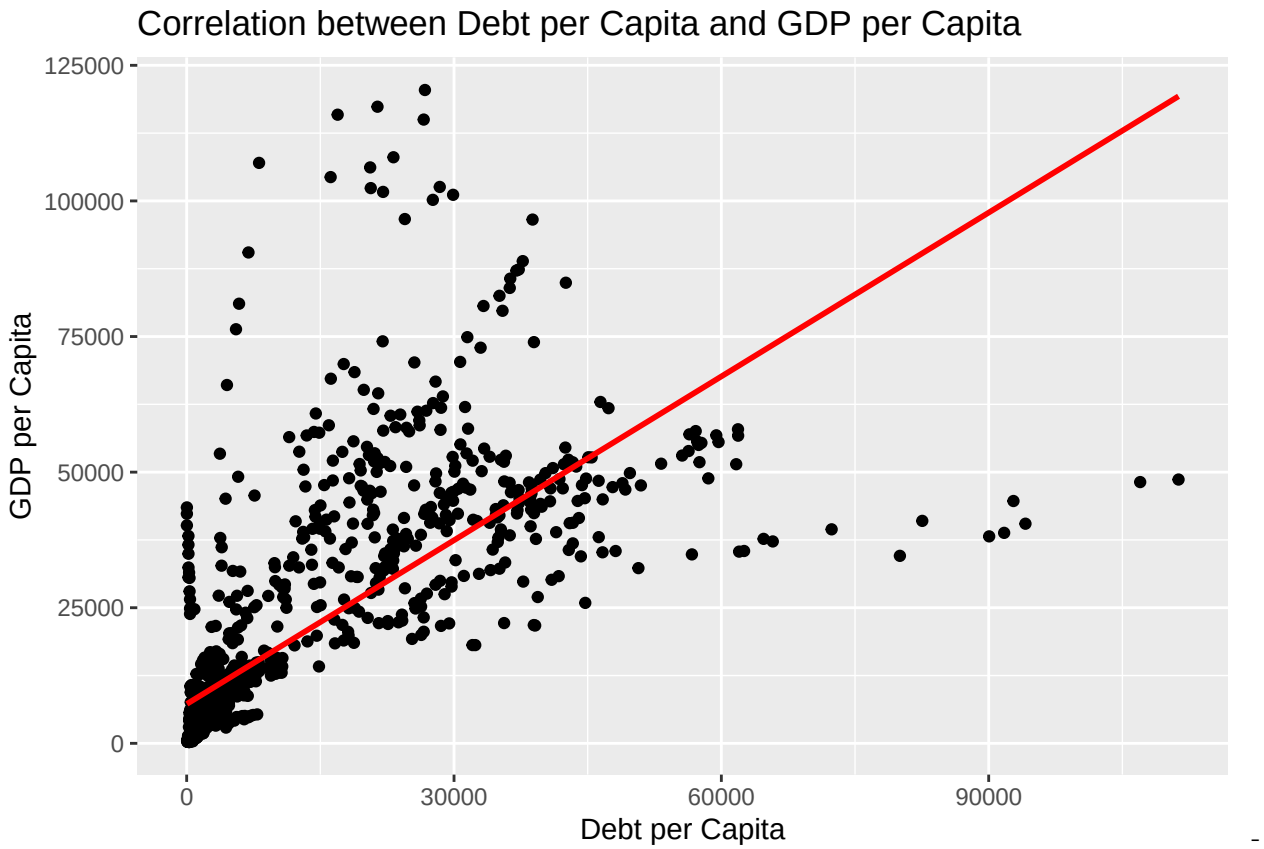
```
# Scatter plot with regression line for Debt vs. GDP
ggplot(country, aes(x = Debt, y = Annual.GDP)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, color = "blue") +
  labs(title = "Correlation between Debt and GDP",
       x = "Debt",
       y = "GDP")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
# Scatter plot with regression line for Debt per capita vs. GDP per capita
ggplot(country, aes(x = Debt.Per.Capita, y = GDP.per.capita)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(title = "Correlation between Debt per Capita and GDP per Capita",
       x = "Debt per Capita",
       y = "GDP per Capita")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




Correlation appears to decrease as all values increase - Example: GDP.per.capita and Debt.per.capita become less linear as the values increase

## Random Data Visualizations

```
# Load necessary libraries
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

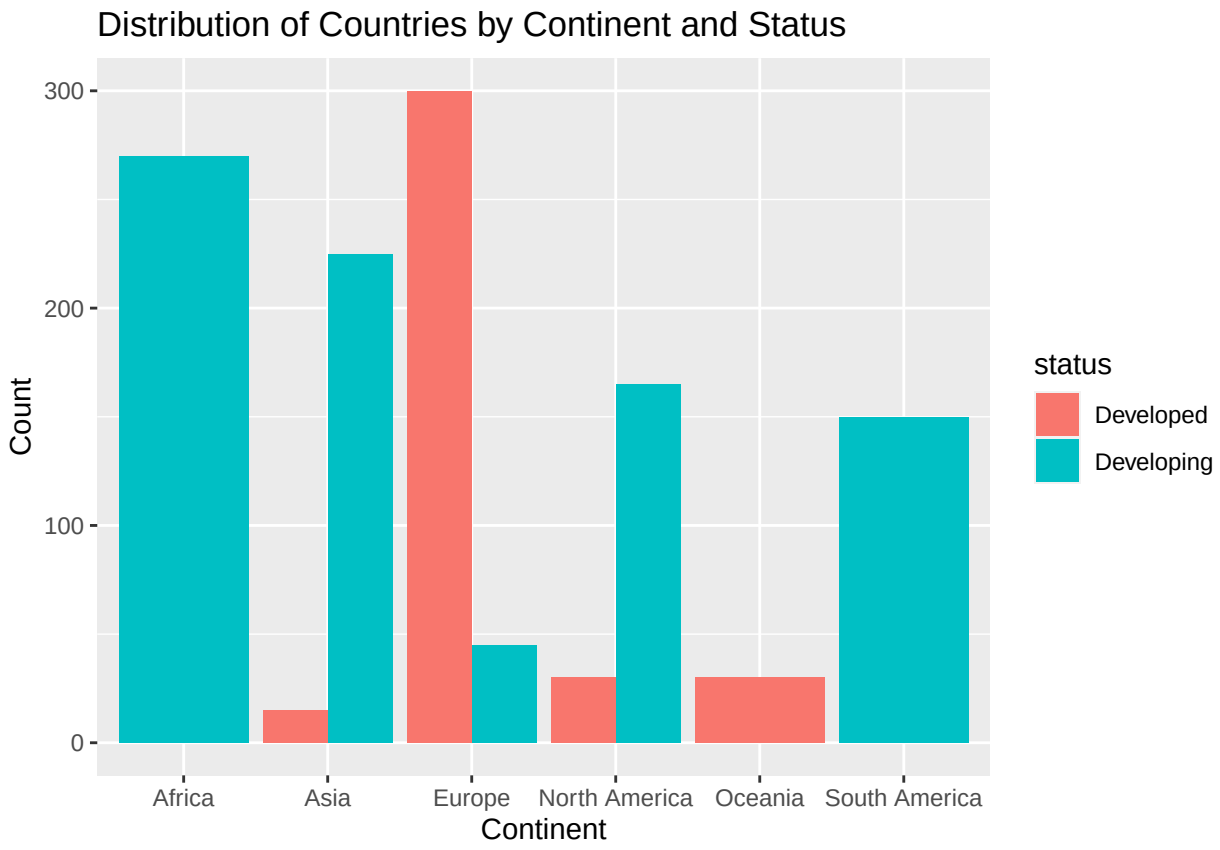
```
library(car) # For MANOVA
```

```
## Loading required package: carData
```

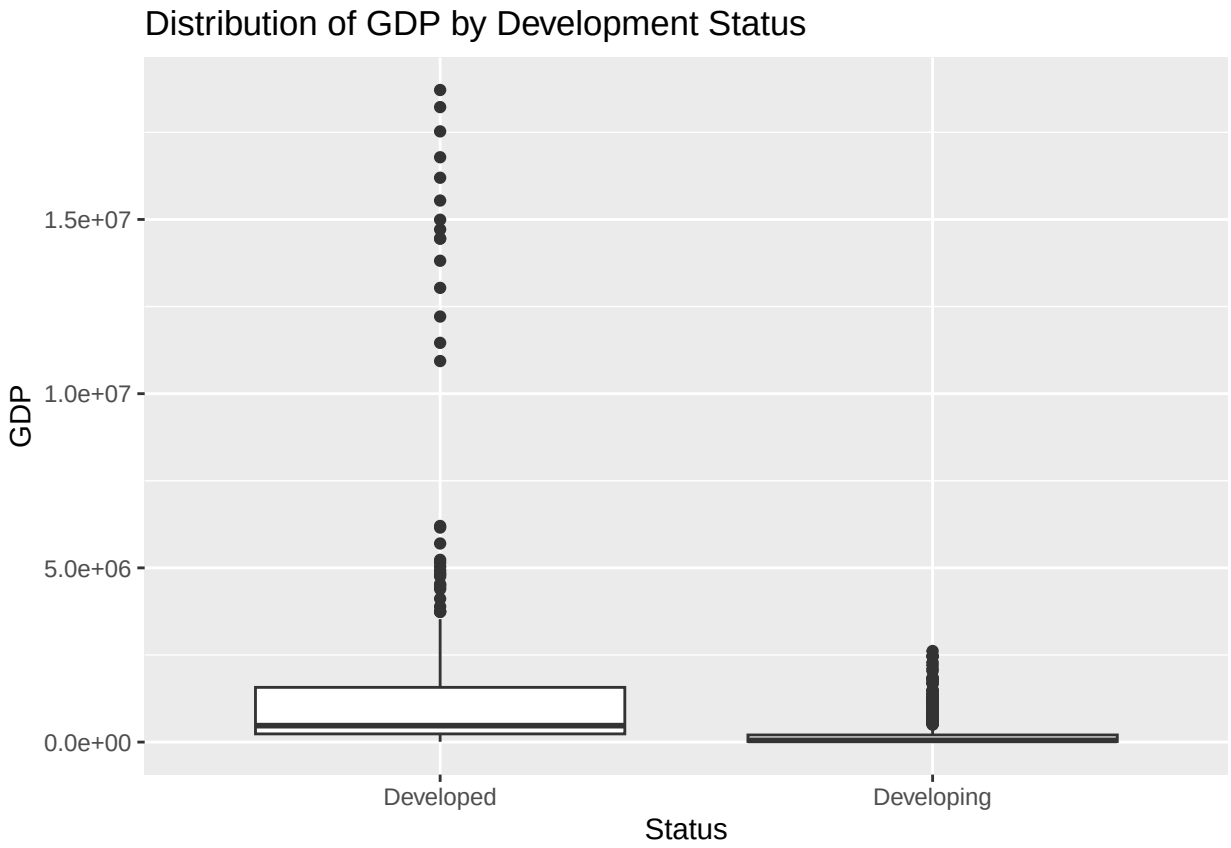
```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

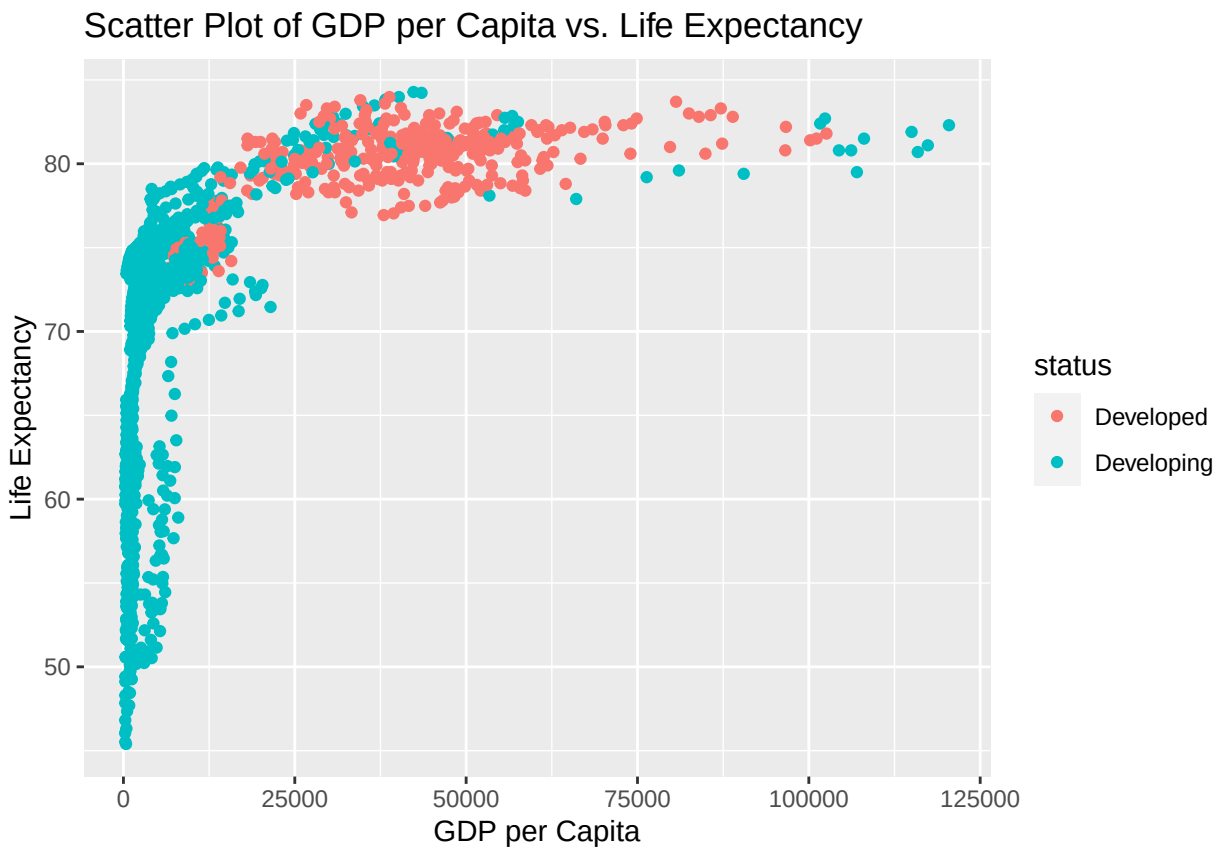
```r
#library(FactoMineR) # For PCA
library(stats) # For normality tests

ggplot(country, aes(x = Continent, fill = status)) +
  geom_bar(position = "dodge") +
  labs(title = "Distribution of Countries by Continent and Status",
       x = "Continent", y = "Count")
```



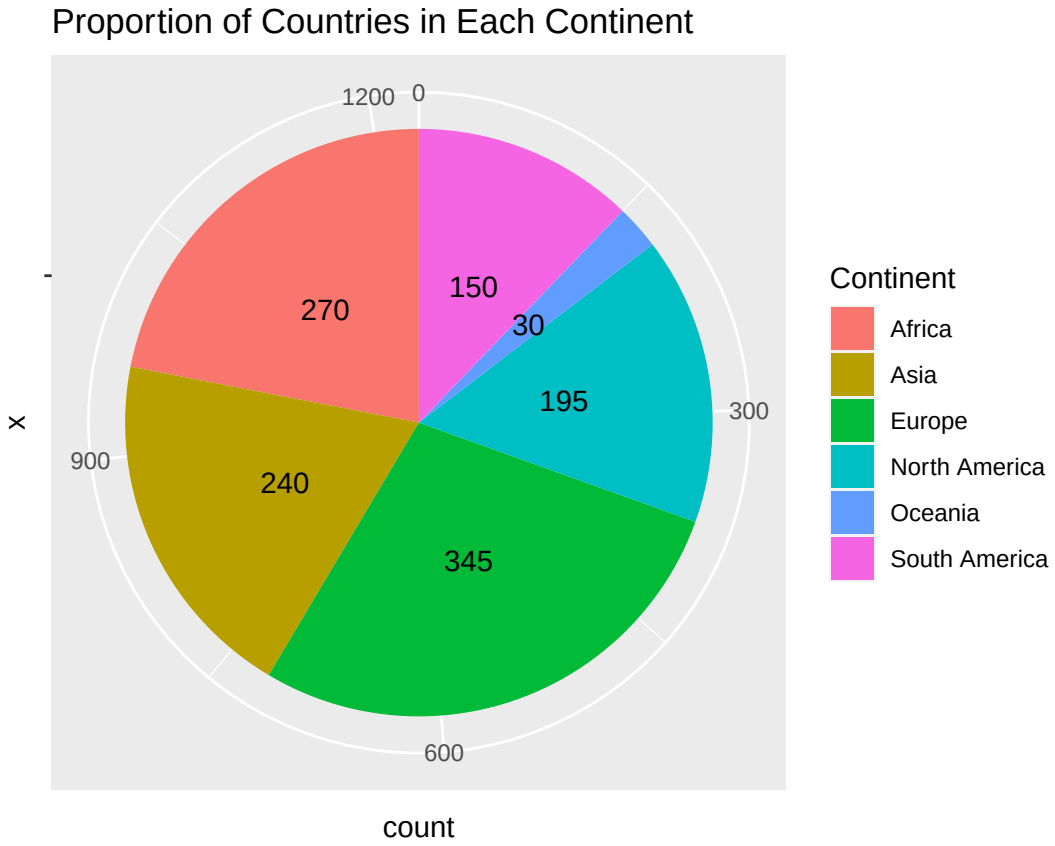

```r
ggplot(country, aes(x = status, y = Annual.GDP)) +
  geom_boxplot() +
  labs(title = "Distribution of GDP by Development Status",
       x = "Status", y = "GDP")
```



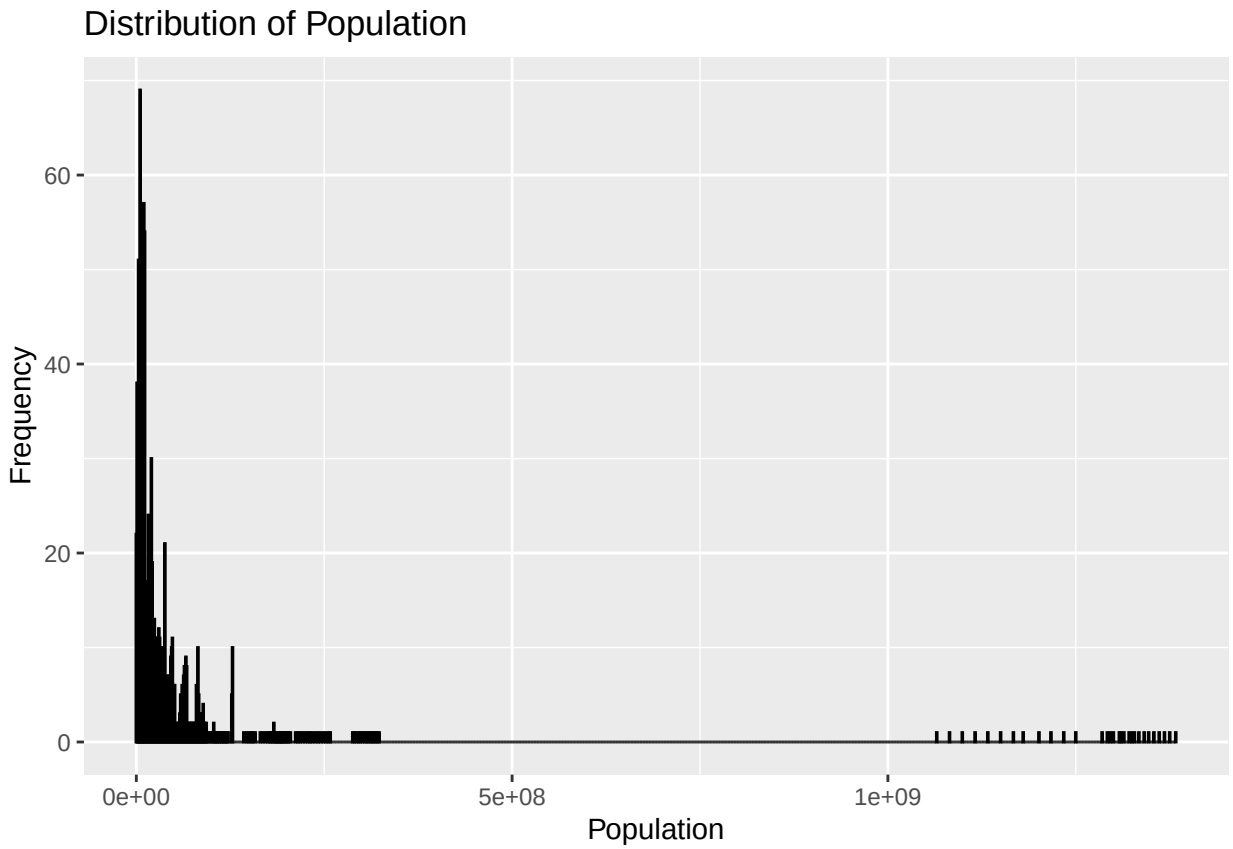

```
ggplot(country, aes(x = GDP.per.capita, y = Life.expectancy, color=status)) +
  geom_point() +
  labs(title = "Scatter Plot of GDP per Capita vs. Life Expectancy",
       x = "GDP per Capita", y = "Life Expectancy")
```




```r
# Create a new column for counting
country.count <- country %>%
  group_by(Continent) %>%
  summarise(count = n())

# Plot the polar bar chart with counts
ggplot(country.count, aes(x = "", y = count, fill = Continent)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  geom_text(aes(label = count), position = position_stack(vjust = 0.5)) +
  labs(title = "Proportion of Countries in Each Continent")
```

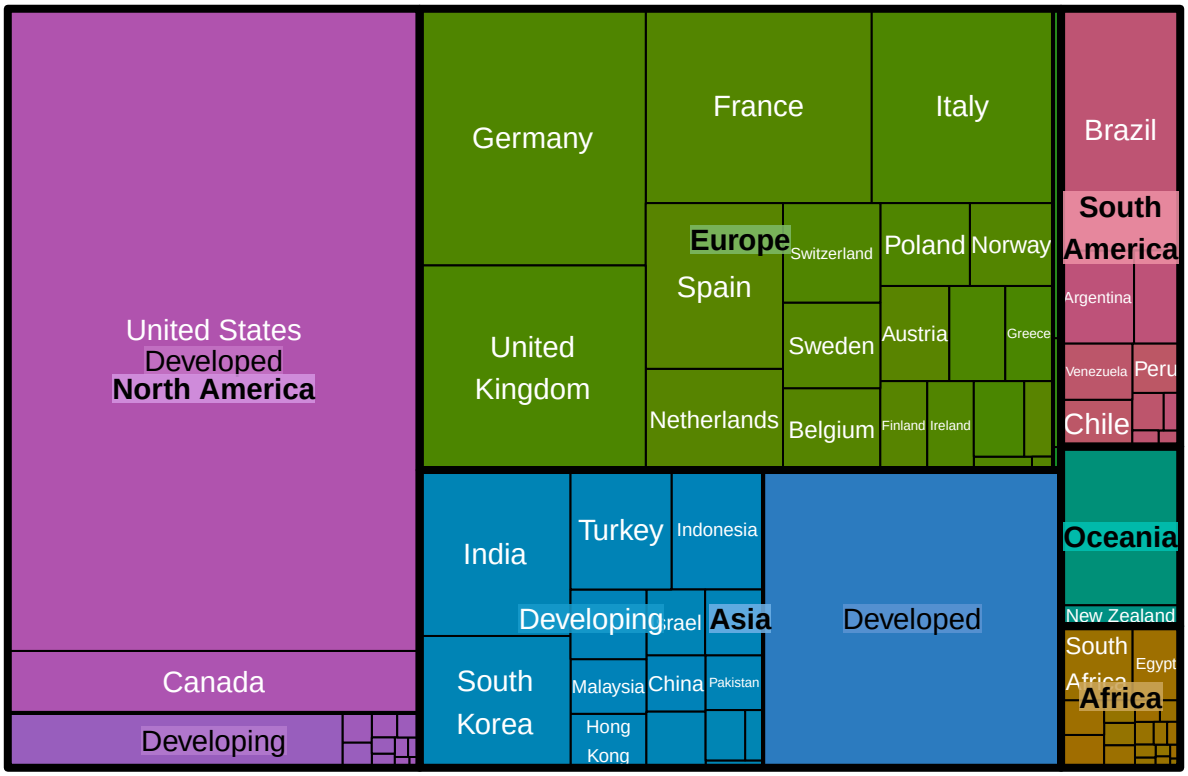



```
ggplot(country, aes(x = Population)) +
  geom_histogram(binwidth = 1000000, fill = "skyblue", color = "black", alpha = 0.7) +
  labs(title = "Distribution of Population",
       x = "Population", y = "Frequency")
```

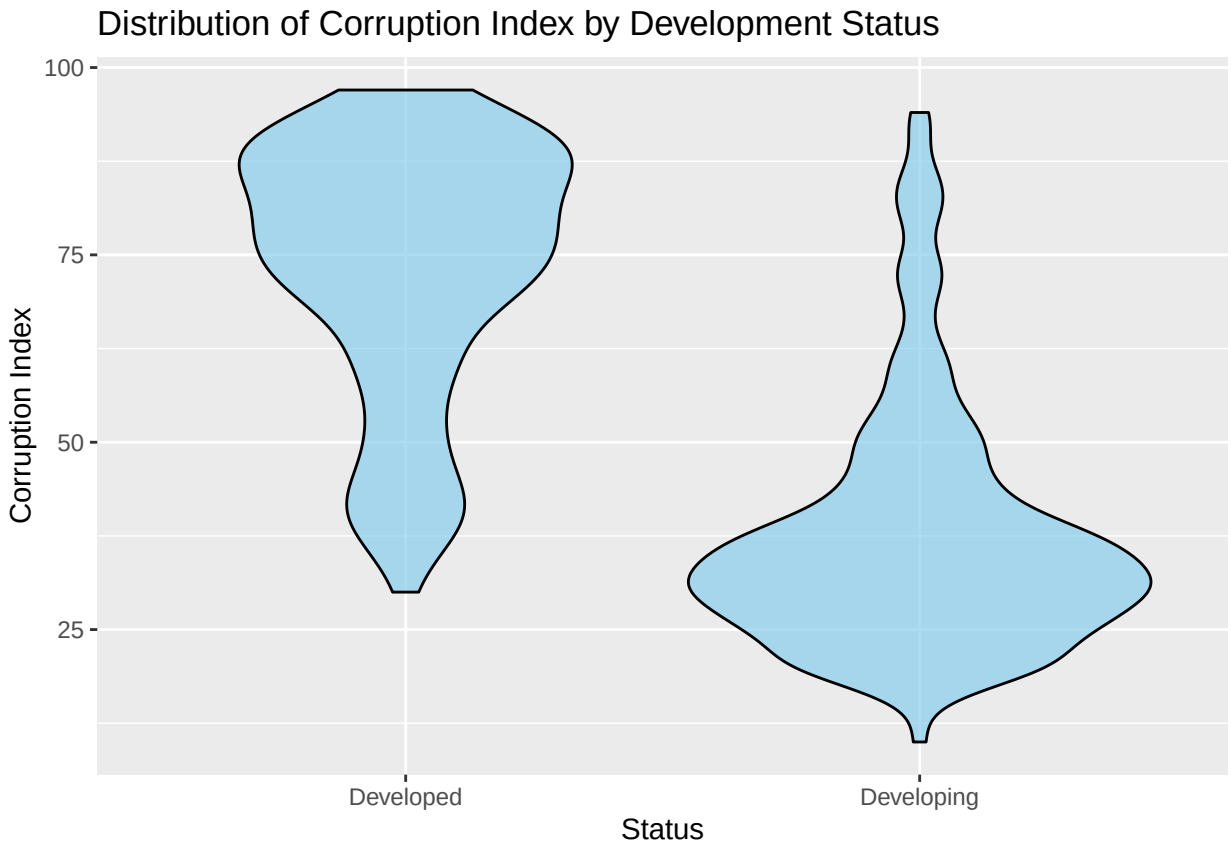



```
library(treemap)
treemap(
  dtf = country,
  index = c("Continent", "status", "Country"),  # Hierarchical index
  vSize = "Annual.GDP",
  vColor = "Annual.GDP",
  draw = TRUE  # Set to TRUE to display the treemap
)
```

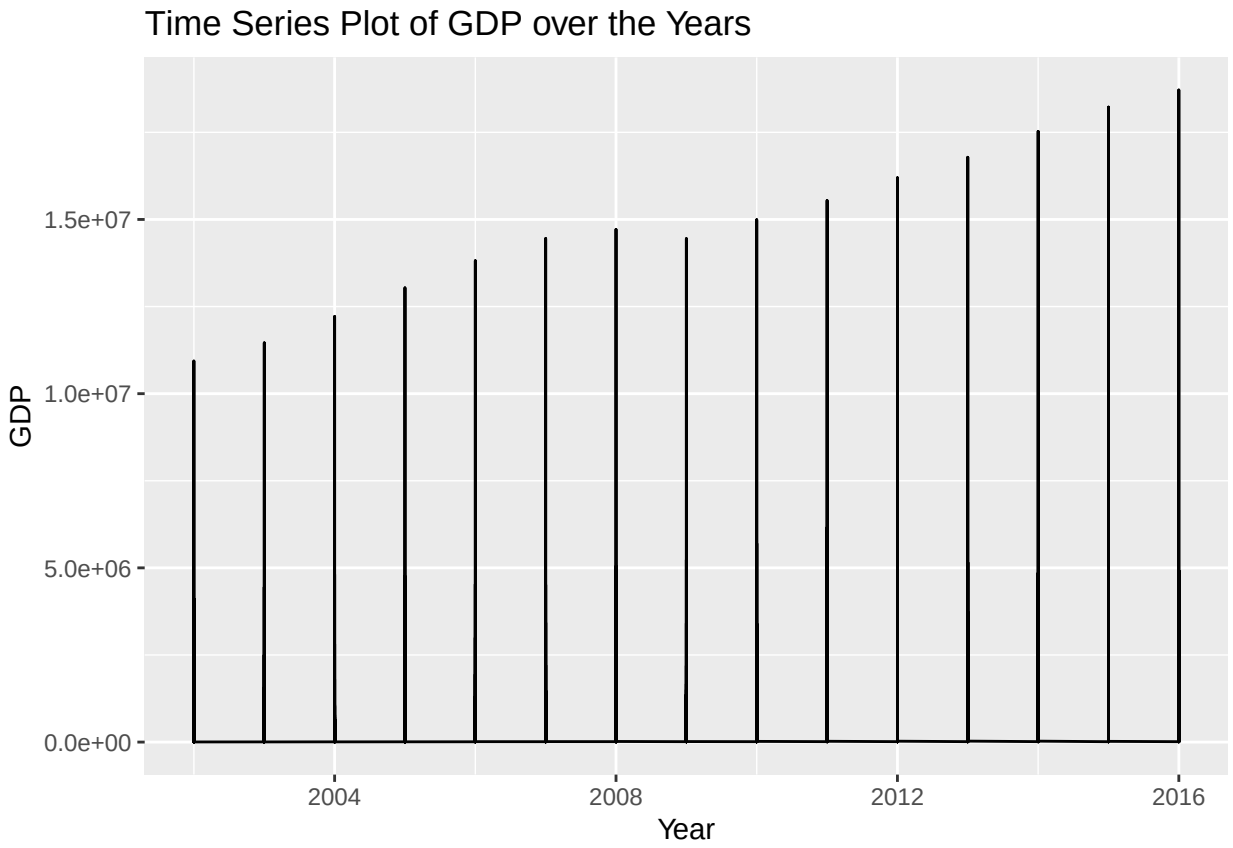
Annual.GDP




```r
# 8. Violin Plot for Distribution of a Continuous Variable
ggplot(country, aes(x = status, y = Corruption.Index)) +
  geom_violin(fill = "skyblue", color = "black", alpha = 0.7) +
  labs(title = "Distribution of Corruption Index by Development Status",
       x = "Status", y = "Corruption Index")
```




```
ggplot(country, aes(x = Date, y = Annual.GDP)) +
  geom_line() +
  labs(title = "Time Series Plot of GDP over the Years",
       x = "Year", y = "GDP")
```




## Economic Analysis - Let's evaluate the developed and developing countries based on their economic variables

### Status Means

```
# Calculate group mean
tapply(country[, 5:22], country$status, colMeans)
```

```
## $Developed
##              Annual.GDP         GDP.per.capita                   Debt
##           1.556960e+06           3.989115e+04           1.458779e+06
##        Debt.Per.Capita                Deficit            Expenditure
##           2.681366e+04          -6.725619e+04           6.395632e+05
## Expenditure.Per.Capita       Corruption.Index                Exports
##           1.769253e+04           7.323200e+01           2.965438e+05
##          Exports...GDP                Imports          Imports...GDP
##           3.227608e-01           3.231595e+05           3.235037e-01
##             Population         Fertility.Rate       Crude.death.rate
##           3.828944e+07           1.646907e+00           9.149040e-03
##                    HDI        Life.expectancy     CO2.Tons.per.capita
##           8.833867e-01           7.998704e+01           9.402587e+00
##
## $Developing
##              Annual.GDP         GDP.per.capita                   Debt
##           1.949453e+05           7.949827e+03           1.285428e+05
##        Debt.Per.Capita                Deficit            Expenditure
##           3.167475e+03          -1.310582e+06           8.046727e+06
```

```
## Expenditure.Per.Capita       Corruption.Index               Exports
##           3.155549e+05           3.768889e+01          8.059101e+04
##          Exports...GDP                Imports          Imports...GDP
##           3.105477e-01           8.001617e+04          3.783656e-01
##             Population         Fertility.Rate       Crude.death.rate
##           7.701842e+07           2.950573e+00          7.340433e-03
##                    HDI        Life.expectancy    CO2.Tons.per.capita
##           6.662573e-01           6.970330e+01          3.360749e+00
```

### Inference for Means

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v stringr   1.5.0
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.0
## v readr     2.1.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x car::recode()   masks dplyr::recode()
## x purrr::some()   masks car::some()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(DescTools)
```

```
##
## Attaching package: 'DescTools'
##
## The following object is masked from 'package:car':
##
##     Recode
```

```
library(dplyr)
library(car)


# We want to assess all potential economic factors
economic_vars <- country[, c("Annual.GDP", "GDP.per.capita", "Debt", "Expenditure", "Exports", "Imports

# Split the data based on the 'Status' variable (developed vs. developing)
developed_data <- economic_vars[country$status == "Developed", ]
developing_data <- economic_vars[country$status == "Developing", ]

# Perform Hotelling's T-squared test
HotellingsT2Test(developed_data, developing_data)
```

```
##
##  Hotelling's two sample T2-test
##
## data:  developed_data and developing_data
## T.2 = 191.13, df1 = 6, df2 = 1223, p-value < 2.2e-16
## alternative hypothesis: true location difference is not equal to c(0,0,0,0,0,0)
```

- T2: 191.133

- P-value: <2.2e-16
- Null Hypothesis (H0): There is no difference in the mean economic profiles (GDP, GDP per capita, Debt, Expenditure, Exports, Imports) between developed and developing countries.
- Alternative Hypothesis (H1): There is a significant difference in the mean economic profiles between developed and developing countries.
- reject the null hypothesis. there are significant differences in the mean economic profiles between developed and developing countries.

### MANOVA

```
fit.lm <- lm(cbind(Annual.GDP, GDP.per.capita, Expenditure, Debt, Exports, Imports) ~ status, data = cou
# Display the results
summary(Manova(fit.lm))
```

```
##
## Type II MANOVA Tests:
##
## Sum of squares and products for error:
##                   Annual.GDP GDP.per.capita   Expenditure         Debt
## Annual.GDP     3.510649e+15   3.114306e+12  5.600424e+14 3.396201e+15
## GDP.per.capita 3.114306e+12   3.169440e+11 -2.900055e+13 2.421890e+12
## Expenditure    5.600424e+14  -2.900055e+13  2.322602e+19 1.051134e+15
## Debt           3.396201e+15   2.421890e+12  1.051134e+15 4.057227e+15
## Exports        3.209178e+14   9.694700e+11 -2.022827e+14 3.646965e+14
## Imports        4.727503e+14   9.456398e+11 -2.612316e+14 4.850733e+14
##                      Exports       Imports
## Annual.GDP      3.209178e+14  4.727503e+14
## GDP.per.capita  9.694700e+11  9.456398e+11
## Expenditure    -2.022827e+14 -2.612316e+14
## Debt            3.646965e+14  4.850733e+14
## Exports         8.572376e+13  8.734344e+13
## Imports         8.734344e+13  1.005932e+14
##
## ----------------------------------------
##
## Term: status
##
## Sum of squares and products for the hypothesis:
##                     Annual.GDP GDP.per.capita   Expenditure          Debt
## Annual.GDP      4.835664e+14   1.134037e+13 -2.629821e+15  4.722838e+14
## GDP.per.capita  1.134037e+13   2.659488e+11 -6.167328e+13  1.107577e+13
## Expenditure    -2.629821e+15  -6.167328e+13  1.430198e+16 -2.568462e+15
## Debt            4.722838e+14   1.107577e+13 -2.568462e+15  4.612646e+14
## Exports         7.667133e+13   1.798059e+12 -4.169683e+14  7.488245e+13
## Imports         8.632498e+13   2.024452e+12 -4.694686e+14  8.431085e+13
##                      Exports       Imports
## Annual.GDP      7.667133e+13  8.632498e+13
## GDP.per.capita  1.798059e+12  2.024452e+12
## Expenditure    -4.169683e+14 -4.694686e+14
## Debt            7.488245e+13  8.431085e+13
## Exports         1.215654e+13  1.368716e+13
## Imports         1.368716e+13  1.541051e+13
##
## Multivariate Tests: status
```

```
##                   Df test stat approx F num Df den Df     Pr(>F)
## Pillai             1 0.4839135 191.1263      6   1223 < 2.22e-16 ***
## Wilks              1 0.5160865 191.1263      6   1223 < 2.22e-16 ***
## Hotelling-Lawley   1 0.9376597 191.1263      6   1223 < 2.22e-16 ***
## Roy                1 0.9376597 191.1263      6   1223 < 2.22e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- F-Value: 376.6468
- DF: (3,1226)
- P-value: < 2.22e-16
- Null Hypothesis (H0): There is no significant difference in the means of the dependent variables across the "status" groups.
- Alternative Hypothesis (H1): There is a significant difference in the means of the dependent variables across the "status" groups.
- Wilks P-value is less than 2.22e-16, reject the null hypothesis. There is strong evidence that the means of the dependent variables (e.g., GDP, GDP per capita, Expenditure) differ significantly across the "status" groups (developed and developing countries)

**Test for multivariate Normality**

```
developed_data <- country[country$status == "Developed", ]
developing_data <- country[country$status == "Developing", ]

library(mvShapiroTest)
mvShapiro.Test(as.matrix(developed_data[,5:22]))
```

```
##
##  Generalized Shapiro-Wilk test for Multivariate Normality by
##  Villasenor-Alva and Gonzalez-Estrada
##
## data:  as.matrix(developed_data[, 5:22])
## MVW = 0.8739, p-value < 2.2e-16
```

```
mvShapiro.Test(as.matrix(developing_data[,5:22]))
```

```
##
##  Generalized Shapiro-Wilk test for Multivariate Normality by
##  Villasenor-Alva and Gonzalez-Estrada
##
## data:  as.matrix(developing_data[, 5:22])
## MVW = 0.60412, p-value < 2.2e-16
```

- The extremely low p-value indicates strong evidence against the null hypothesis that the data in the developed countries group follows a multivariate normal distribution.

## Corruption Analaysis

```
correlations <- cor(country[, 5:22])
correlations
```

```
##                        Annual.GDP GDP.per.capita         Debt Debt.Per.Capita
## Annual.GDP          1.000000000    0.299569867  0.910601651      0.48877956
## GDP.per.capita      0.299569867    1.000000000  0.263007105      0.72100570
## Debt                0.910601651    0.263007105  1.000000000      0.58969114
## Debt.Per.Capita     0.488779559    0.721005704  0.589691141      1.00000000
```

```
## Deficit                 0.001080275    0.005817376  0.003398086      0.01195927
## Expenditure            -0.006793397   -0.024635794 -0.004682325     -0.01519959
## Expenditure.Per.Capita -0.011169521   -0.024524094 -0.008936327     -0.01647295
## Corruption.Index        0.253232534    0.822905127  0.215156265      0.63550340
## Exports                 0.635874084    0.366395569  0.660986332      0.48247765
## Exports...GDP          -0.131377739    0.211256695 -0.133425323      0.12329563
## Imports                 0.821331789    0.361193202  0.786452912      0.49070541
## Imports...GDP          -0.178954170    0.068048132 -0.177585718     -0.01069346
## Population              0.195556688   -0.085113537  0.244810958     -0.04612631
## Fertility.Rate         -0.192458565   -0.484982031 -0.180397835     -0.42927348
## Crude.death.rate        0.046339649    0.038981623  0.068603345      0.10511149
## HDI                     0.296429902    0.760408695  0.255577752      0.64770303
## Life.expectancy         0.223207238    0.621177161  0.210044409      0.55329128
## CO2.Tons.per.capita     0.363423071    0.696580631  0.304306786      0.50286341
##                              Deficit   Expenditure Expenditure.Per.Capita
## Annual.GDP              0.0010802746 -0.0067933969          -0.0111695212
## GDP.per.capita          0.0058173757 -0.0246357937          -0.0245240938
## Debt                    0.0033980860 -0.0046823246          -0.0089363268
## Debt.Per.Capita         0.0119592722 -0.0151995942          -0.0164729475
## Deficit                 1.0000000000 -0.0626585802          -0.0634011366
## Expenditure            -0.0626585802  1.0000000000           0.9999557442
## Expenditure.Per.Capita -0.0634011366  0.9999557442           1.0000000000
## Corruption.Index        0.0369408828 -0.0402495916          -0.0401713634
## Exports                 0.0021499215 -0.0129836865          -0.0165026525
## Exports...GDP           0.0029957699 -0.0022549981          -0.0014015194
## Imports                 0.0048630560 -0.0140728548          -0.0182350225
## Imports...GDP           0.0153237733 -0.0396139582          -0.0386766969
## Population              0.0036510335 -0.0064213023          -0.0083440691
## Fertility.Rate          0.0009680047  0.0048172341           0.0051472425
## Crude.death.rate        0.0216248576 -0.0422134871          -0.0423386775
## HDI                    -0.0062676791 -0.0108026336          -0.0110987015
## Life.expectancy        -0.0033794500  0.0006695299           0.0004558625
## CO2.Tons.per.capita    -0.0042694905  0.0028572203           0.0021443102
##                        Corruption.Index      Exports Exports...GDP      Imports
## Annual.GDP                   0.25323253  0.635874084  -0.131377739  0.821331789
## GDP.per.capita               0.82290513  0.366395569   0.211256695  0.361193202
## Debt                         0.21515627  0.660986332  -0.133425323  0.786452912
## Debt.Per.Capita              0.63550340  0.482477652   0.123295628  0.490705412
## Deficit                      0.03694088  0.002149922   0.002995770  0.004863056
## Expenditure                 -0.04024959 -0.012983686  -0.002254998 -0.014072855
## Expenditure.Per.Capita      -0.04017136 -0.016502652  -0.001401519 -0.018235022
## Corruption.Index             1.00000000  0.328306564   0.299027298  0.321982893
## Exports                      0.32830656  1.000000000   0.125498692  0.948133191
## Exports...GDP                0.29902730  0.125498692   1.000000000  0.068021011
## Imports                      0.32198289  0.948133191   0.068021011  1.000000000
## Imports...GDP                0.17974910  0.013895577   0.884795669 -0.011001650
## Population                  -0.11740148  0.494607545  -0.119815163  0.453347686
## Fertility.Rate              -0.55415821 -0.336255929  -0.257773672 -0.311512767
## Crude.death.rate             0.02400498  0.030397423  -0.067826331  0.029386687
## HDI                          0.79928743  0.392554642   0.268674501  0.382709416
## Life.expectancy              0.64451605  0.338058506   0.211546378  0.320967454
## CO2.Tons.per.capita          0.64007189  0.375023343   0.216333069  0.398192865
##                        Imports...GDP   Population Fertility.Rate
## Annual.GDP              -0.17895417  0.195556688  -0.1924585653
```

```
## GDP.per.capita            0.06804813 -0.085113537  -0.4849820314
## Debt                     -0.17758572  0.244810958  -0.1803978351
## Debt.Per.Capita          -0.01069346 -0.046126310  -0.4292734823
## Deficit                   0.01532377  0.003651033   0.0009680047
## Expenditure              -0.03961396 -0.006421302   0.0048172341
## Expenditure.Per.Capita   -0.03867670 -0.008344069   0.0051472425
## Corruption.Index          0.17974910 -0.117401477  -0.5541582053
## Exports                   0.01389558  0.494607545  -0.3362559288
## Exports...GDP             0.88479567 -0.119815163  -0.2577736722
## Imports                  -0.01100165  0.453347686  -0.3115127666
## Imports...GDP             1.00000000 -0.162040565  -0.2129311535
## Population               -0.16204057  1.000000000  -0.0655877212
## Fertility.Rate           -0.21293115 -0.065587721   1.0000000000
## Crude.death.rate         -0.14576248 -0.055289081   0.2419994643
## HDI                       0.14089755 -0.097603505  -0.8402881315
## Life.expectancy           0.16808310 -0.036182005  -0.8601687950
## CO2.Tons.per.capita       0.05485838 -0.020036280  -0.5329793800
##                       Crude.death.rate          HDI Life.expectancy
## Annual.GDP                  0.04633965  0.296429902    0.2232072381
## GDP.per.capita              0.03898162  0.760408695    0.6211771609
## Debt                        0.06860334  0.255577752    0.2100444085
## Debt.Per.Capita             0.10511149  0.647703031    0.5532912787
## Deficit                     0.02162486 -0.006267679   -0.0033794500
## Expenditure                -0.04221349 -0.010802634    0.0006695299
## Expenditure.Per.Capita     -0.04233868 -0.011098702    0.0004558625
## Corruption.Index            0.02400498  0.799287433    0.6445160467
## Exports                     0.03039742  0.392554642    0.3380585057
## Exports...GDP              -0.06782633  0.268674501    0.2115463782
## Imports                     0.02938669  0.382709416    0.3209674536
## Imports...GDP              -0.14576248  0.140897548    0.1680831035
## Population                 -0.05528908 -0.097603505   -0.0361820052
## Fertility.Rate              0.24199946 -0.840288131   -0.8601687950
## Crude.death.rate            1.00000000 -0.164938738   -0.4337332930
## HDI                        -0.16493874  1.000000000    0.9060353597
## Life.expectancy            -0.43373329  0.906035360    1.0000000000
## CO2.Tons.per.capita         0.02034726  0.691908246    0.5287501612
##                       CO2.Tons.per.capita
## Annual.GDP                    0.363423071
## GDP.per.capita                0.696580631
## Debt                          0.304306786
## Debt.Per.Capita               0.502863415
## Deficit                      -0.004269491
## Expenditure                   0.002857220
## Expenditure.Per.Capita        0.002144310
## Corruption.Index              0.640071888
## Exports                       0.375023343
## Exports...GDP                 0.216333069
## Imports                       0.398192865
## Imports...GDP                 0.054858382
## Population                   -0.020036280
## Fertility.Rate               -0.532979380
## Crude.death.rate              0.020347257
## HDI                           0.691908246
## Life.expectancy               0.528750161
```

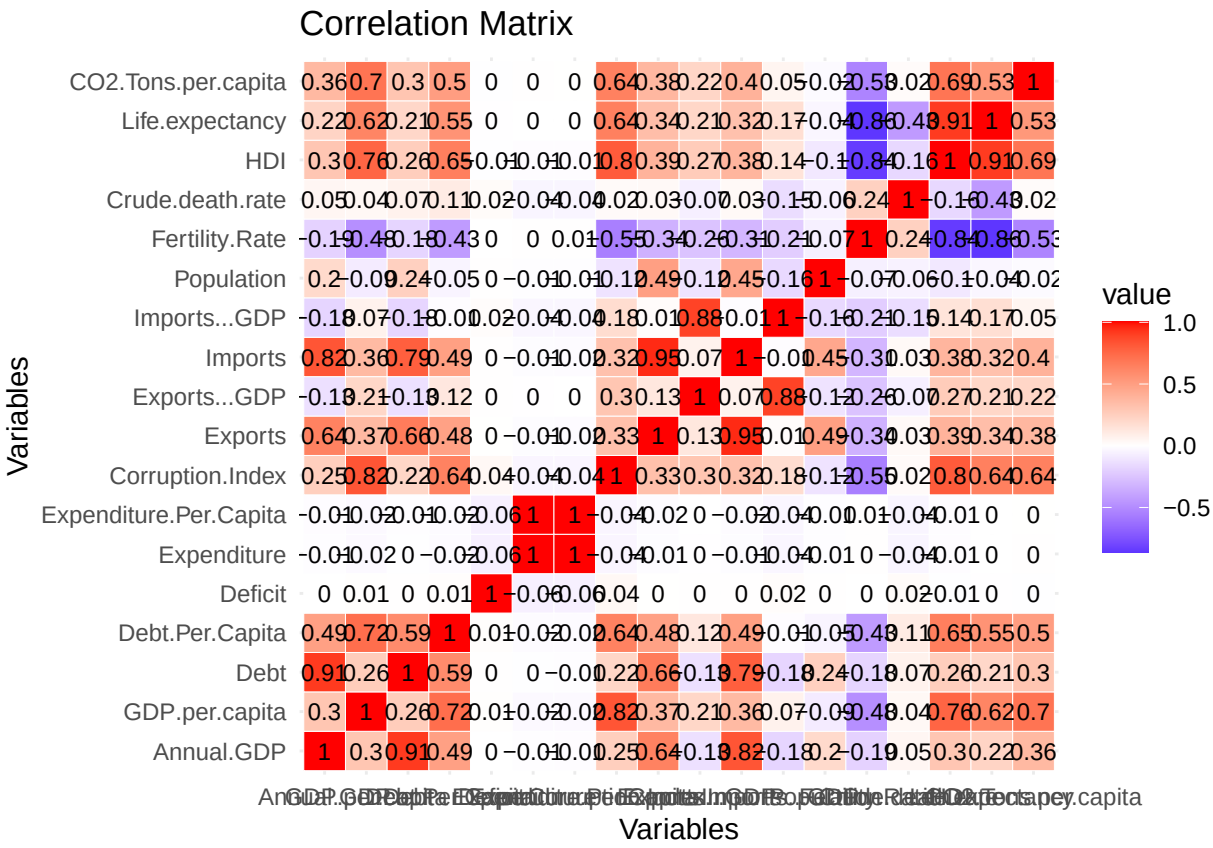
```
## CO2.Tons.per.capita             1.000000000
```

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
# Melt the correlation matrix for ggplot
melted_corr <- melt(correlations)

# Plot using ggplot2 with numbers
ggplot(melted_corr, aes(Var1, Var2, fill = value, label = round(value, 2))) +
  geom_tile(color = "white") +
  geom_text(size = 3) +
  scale_fill_gradient2(low = "blue", mid = "white", high = "red", midpoint = 0) +
  theme_minimal() +
  labs(title = "Correlation Matrix",
       x = "Variables",
       y = "Variables")
```




## Predict and Classify Corruption Index

- Using variables that I see highly correlated with these variables

### MAKE COUNTRY CONTINENT

```
country$Country <- as.factor(country$Country)
country$Continent <- as.factor(country$Continent)
```

```
# Load necessary libraries
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following objects are masked from 'package:DescTools':
##
##     MAE, RMSE
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
# Split the data into training and testing sets
set.seed(122)
trainIndex <- createDataPartition(country$Corruption.Index, p = 0.8, list = FALSE)
train_data <- country[trainIndex, ]
test_data <- country[-trainIndex, ]


# Compute correlation matrix
correlation_matrix <- cor(train_data[,5:22])
correlation_with_target <- correlation_matrix[, "Corruption.Index"]

# Display variables sorted by correlation
print(sort(correlation_with_target, decreasing = TRUE))
```

```
##       Corruption.Index         GDP.per.capita                    HDI
##            1.000000000             0.822703698           0.802673306
##        Life.expectancy    CO2.Tons.per.capita        Debt.Per.Capita
##            0.647784040             0.644813806           0.627681522
##                Exports                 Imports          Exports...GDP
##            0.329993806             0.324044116           0.289381481
##             Annual.GDP                    Debt          Imports...GDP
```

```
##            0.257796499            0.221481771            0.175275697
##       Crude.death.rate                Deficit            Expenditure
##            0.015933088            0.005554742           -0.045185472
## Expenditure.Per.Capita             Population        Fertility.Rate
##           -0.045187850           -0.111186316           -0.552794683
```

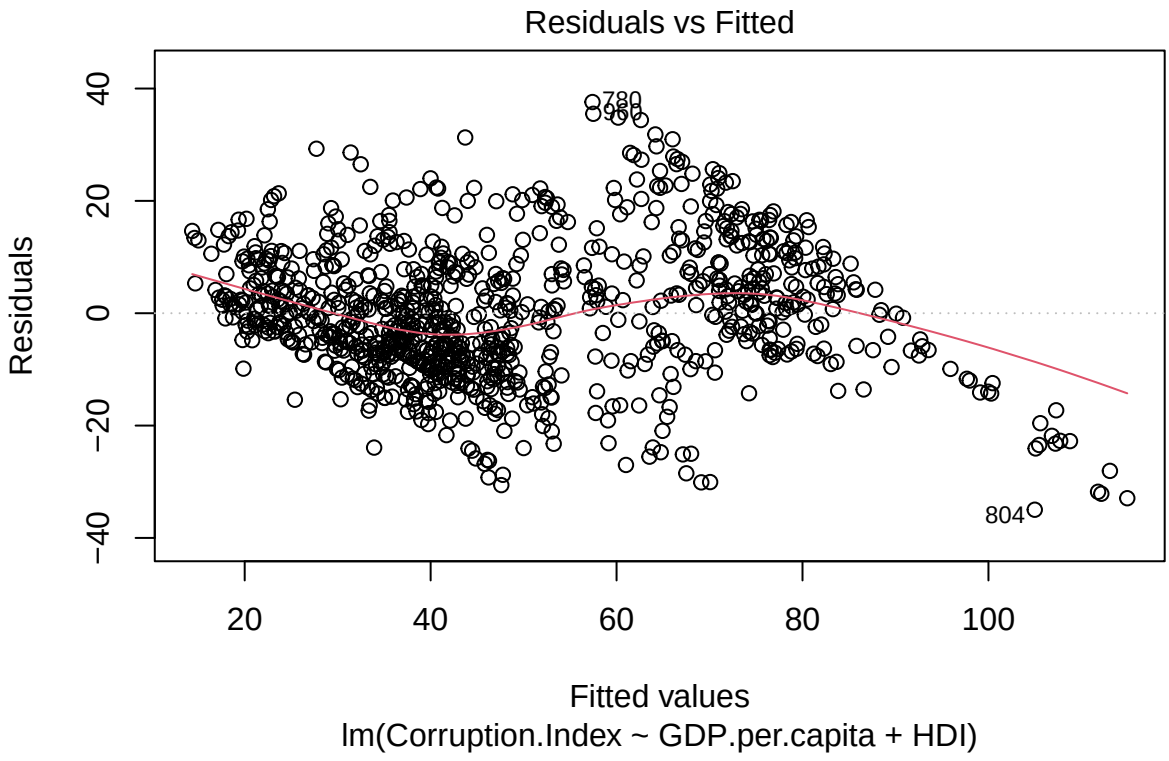
```
# Corruption Index Prediction
corruption_model <- lm(Corruption.Index ~ GDP.per.capita + HDI, data = train_data)

# Make predictions on the test set
corruption_predictions <- predict(corruption_model, newdata = test_data)

# Evaluate the model
corruption_rmse <- sqrt(mean((corruption_predictions - test_data$Corruption.Index)^2, na.rm = TRUE))
print(paste("Corruption Index RMSE: ", corruption_rmse))
```

```
## [1] "Corruption Index RMSE:  11.9012373372759"
```

```
coefficients(corruption_model)
```

```
##    (Intercept) GDP.per.capita            HDI
##  -1.100658e+01   5.393045e-04   6.813699e+01
```

```
# Residual analysis
plot(corruption_model, which = 1)
```




```
summary(corruption_model)
```

```
##
## Call:
## lm(formula = Corruption.Index ~ GDP.per.capita + HDI, data = train_data)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -34.983  -7.389  -0.698   7.112  37.591
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.101e+01  2.577e+00  -4.271 2.13e-05 ***
## GDP.per.capita  5.393e-04  2.598e-05  20.754  < 2e-16 ***
## HDI             6.814e+01  3.924e+00  17.362  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.66 on 982 degrees of freedom
## Multiple R-squared:  0.7527, Adjusted R-squared:  0.7522
## F-statistic:  1495 on 2 and 982 DF,  p-value: < 2.2e-16
```

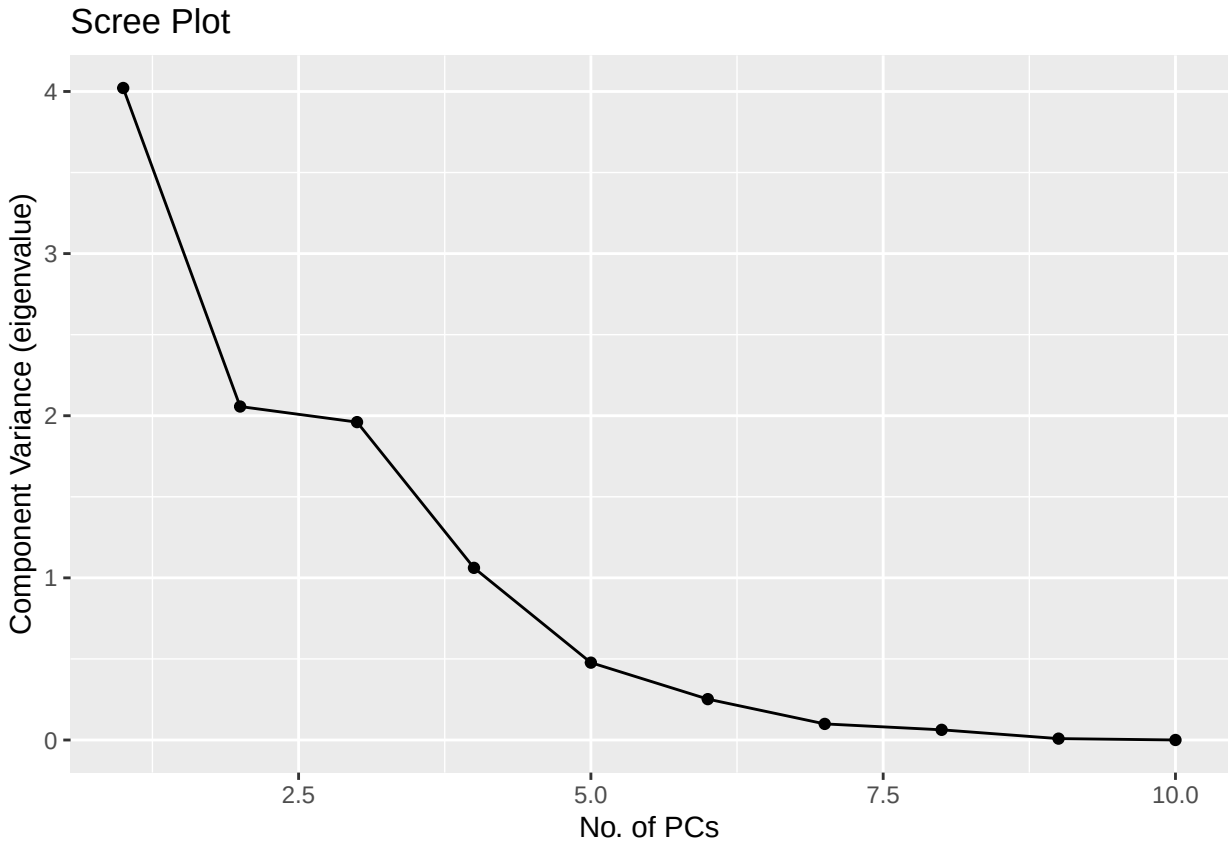
- The R-squared value is 0.7527, suggesting that approximately 75.27% of the variance in the Corruption Index is explained by the model. This is a relatively high value, indicating a good fit.

## Principal Component Analysis on Economic Varibles

```
corruption_dev_vars <- country[, c('Annual.GDP','GDP.per.capita','Debt', 'Debt.Per.Capita','Expenditure
                                   'Expenditure.Per.Capita','Exports', "Exports...GDP", "Imports", "Imp

corruption.pc <- prcomp(corruption_dev_vars, center=T, scale.=T)
summary(corruption.pc)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5     PC6     PC7
## Standard deviation     2.0053 1.4341 1.4002 1.0304 0.69071 0.50205 0.31499
## Proportion of Variance 0.4021 0.2057 0.1961 0.1062 0.04771 0.02521 0.00992
## Cumulative Proportion  0.4021 0.6078 0.8039 0.9100 0.95775 0.98296 0.99288
##                           PC8     PC9     PC10
## Standard deviation     0.25052 0.09183 0.005703
## Proportion of Variance 0.00628 0.00084 0.000000
## Cumulative Proportion  0.99915 1.00000 1.000000
```

```
ggplot(data.frame(x = 1:length(corruption.pc$sdev), y = corruption.pc$sdev^2), aes(x, y)) +
geom_line() +
geom_point() +
labs(x = "No. of PCs", y = "Component Variance (eigenvalue)", title = "Scree Plot")
```




- The first four principal components (PC1 to PC4) capture around 91% of the total variance

```
# Extract loadings
loadings <- corruption.pc$rotation

# Display loadings
print("Loadings:")
```

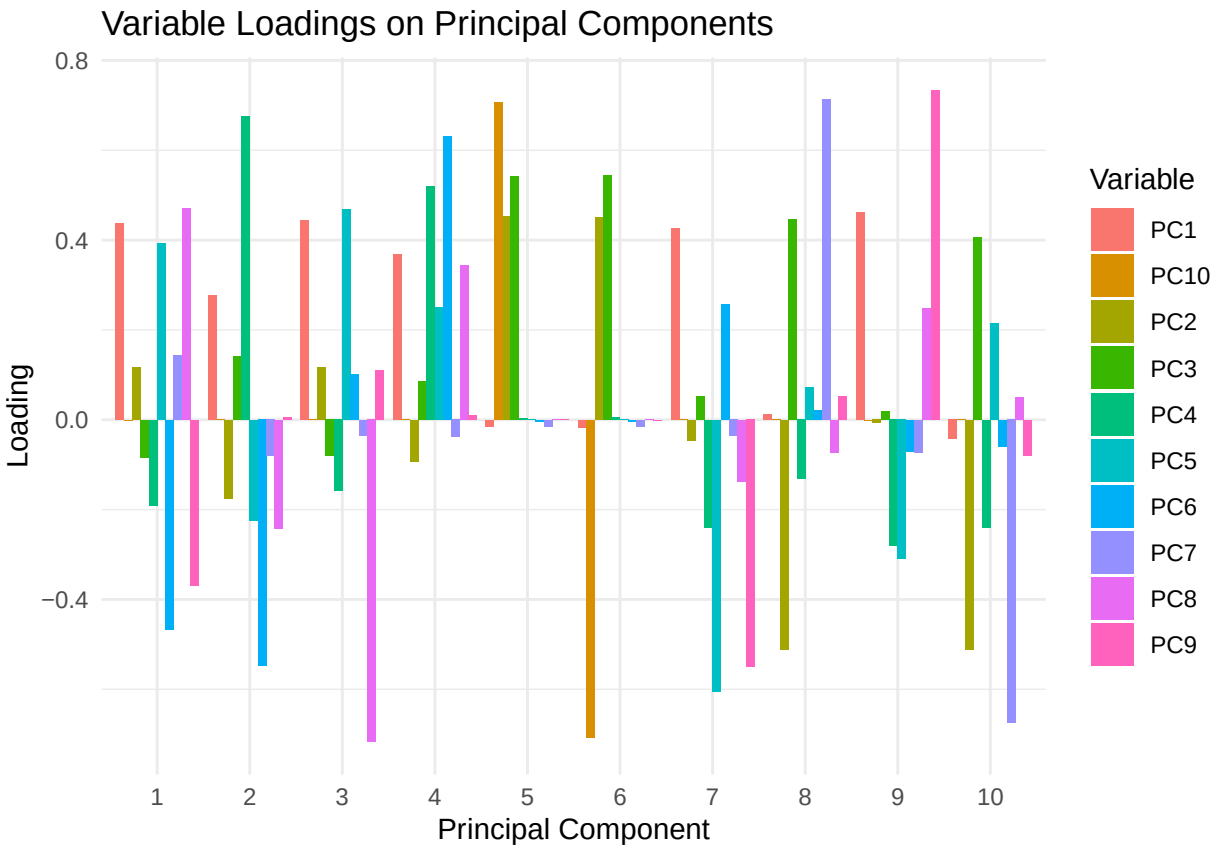
```
## [1] "Loadings:"
```

```
print(loadings)
```

```
##                                  PC1          PC2         PC3          PC4
## Annual.GDP                0.43848904  0.117407612 -0.08424432 -0.191191533
## GDP.per.capita            0.27701835 -0.175201026  0.14243277  0.675459409
## Debt                      0.44394129  0.116640964 -0.08099796 -0.158802817
## Debt.Per.Capita           0.36776014 -0.092400413  0.08560541  0.519844622
## Expenditure              -0.01472011  0.452244456  0.54315674  0.003390623
## Expenditure.Per.Capita -0.01663562  0.451448903  0.54372585  0.005843872
## Exports                   0.42720697 -0.045824623  0.05266465 -0.240442193
## Exports...GDP             0.01123988 -0.510982873  0.44628061 -0.132224342
## Imports                   0.46224394 -0.007222402  0.01958293 -0.281111484
## Imports...GDP            -0.04314848 -0.511654559  0.40711959 -0.239538267
##                                  PC5          PC6         PC7          PC8
## Annual.GDP                0.3933778037 -0.466481380  0.14366277  0.469841939
## GDP.per.capita           -0.2244063021 -0.548223735 -0.08033747 -0.242165170
## Debt                      0.4685331770  0.101500316 -0.03465866 -0.717271005
## Debt.Per.Capita           0.2494236402  0.630049189 -0.03838162  0.343355716
## Expenditure               0.0002399091 -0.003999904 -0.01471702  0.001637298
```

```
## Expenditure.Per.Capita -0.0005483898 -0.003598653 -0.01509246  0.001058933
## Exports                -0.6048518071  0.256172751 -0.03577322 -0.137071282
## Exports...GDP           0.0727272184  0.020585198  0.71289701 -0.073147434
## Imports                -0.3098505392 -0.070729703 -0.07256316  0.248291655
## Imports...GDP           0.2145170228 -0.059175125 -0.67455235  0.049417699
##                                  PC9          PC10
## Annual.GDP             -0.3695243924 -1.348175e-03
## GDP.per.capita          0.0051302502  1.038697e-03
## Debt                    0.1096653127 -9.817701e-04
## Debt.Per.Capita         0.0103606329  3.923180e-04
## Expenditure            -0.0002756236  7.071012e-01
## Expenditure.Per.Capita -0.0012679003 -7.071078e-01
## Exports                -0.5506994103 -8.698106e-05
## Exports...GDP           0.0532429172 -8.950137e-06
## Imports                 0.7341284781 -1.546535e-03
## Imports...GDP          -0.0789976322  1.725069e-04
```

```r
# Visualization of loadings
library(ggplot2)
library(tidyr)

# Convert loadings to a data frame for plotting
loadings_df <- as.data.frame(loadings)
loadings_df$PC <- factor(1:ncol(loadings_df))

# Reshape data for plotting
loadings_long <- gather(loadings_df, key = "Variable", value = "Loading", -PC)

# Plot loadings
ggplot(loadings_long, aes(x = PC, y = Loading, fill = Variable)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(x = "Principal Component", y = "Loading", title = "Variable Loadings on Principal Components") +
  theme_minimal()
```




- PC1:
- Positive Loadings: Annual.GDP, GDP.per.capita, Debt, Debt.Per.Capita, Exports, Exports. . . GDP, Imports.
- Interpretation: This component seems to capture overall economic activity and trade.
- PC2:
- Positive Loadings: Expenditure, Expenditure.Per.Capita.
- Negative Loadings: GDP.per.capita, Imports, Imports. . . GDP.
- Interpretation: This component seems to represent government expenditure and its relationship with GDP, imports, and exports.
- PC3:
- Positive Loadings: Expenditure, Expenditure.Per.Capita.
- Negative Loadings: GDP.per.capita, Imports, Imports. . . GDP.
- Interpretation: Similar to PC2, indicating a relationship between government expenditure and GDP.
- PC4:
- Positive Loadings: GDP.per.capita, Imports, Imports. . . GDP.
- Negative Loadings: Debt, Debt.Per.Capita, Exports, Exports. . . GDP.
- Interpretation: This component seems to capture the trade-off between economic development and trade.

We are going to ignore everything after since as previously discussed, 4 PCs summarizes about 91% of the data.

## K-Means Clustering

- Before we begin our K-means clustering we want to find the optimal amount of clusters as well as understand what variables we should be considered in our model
- To determine the cluster count we will be performing the elbow method (The elbow method looks for the point at which adding more clusters does not significantly reduce the WCSS)
- We will use a correlation matrix to determine which variables to use in order to ensure there is no redundancy (highlu correlated variables will not be beneficial). Will consider variables between .5-.7

```
cor(country[,5:22])
```

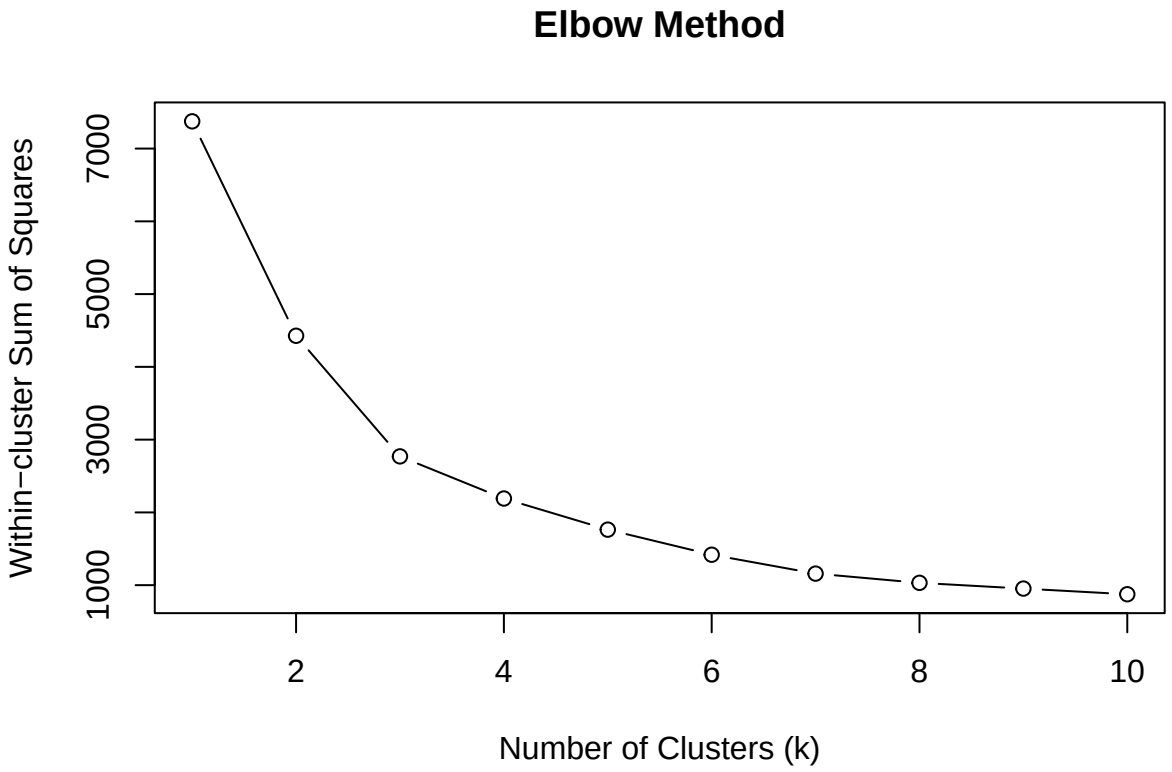
```
##                            Annual.GDP GDP.per.capita         Debt Debt.Per.Capita
## Annual.GDP                1.000000000    0.299569867  0.910601651      0.48877956
## GDP.per.capita            0.299569867    1.000000000  0.263007105      0.72100570
## Debt                      0.910601651    0.263007105  1.000000000      0.58969114
## Debt.Per.Capita           0.488779559    0.721005704  0.589691141      1.00000000
## Deficit                   0.001080275    0.005817376  0.003398086      0.01195927
## Expenditure              -0.006793397   -0.024635794 -0.004682325     -0.01519959
## Expenditure.Per.Capita -0.011169521   -0.024524094 -0.008936327     -0.01647295
## Corruption.Index          0.253232534    0.822905127  0.215156265      0.63550340
## Exports                   0.635874084    0.366395569  0.660986332      0.48247765
## Exports...GDP            -0.131377739    0.211256695 -0.133425323      0.12329563
## Imports                   0.821331789    0.361193202  0.786452912      0.49070541
## Imports...GDP            -0.178954170    0.068048132 -0.177585718     -0.01069346
## Population                0.195556688   -0.085113537  0.244810958     -0.04612631
## Fertility.Rate           -0.192458565   -0.484982031 -0.180397835     -0.42927348
## Crude.death.rate          0.046339649    0.038981623  0.068603345      0.10511149
## HDI                       0.296429902    0.760408695  0.255577752      0.64770303
## Life.expectancy           0.223207238    0.621177161  0.210044409      0.55329128
## CO2.Tons.per.capita       0.363423071    0.696580631  0.304306786      0.50286341
##                                Deficit   Expenditure Expenditure.Per.Capita
## Annual.GDP                0.0010802746 -0.0067933969          -0.0111695212
## GDP.per.capita            0.0058173757 -0.0246357937          -0.0245240938
## Debt                      0.0033980860 -0.0046823246          -0.0089363268
## Debt.Per.Capita           0.0119592722 -0.0151995942          -0.0164729475
## Deficit                   1.0000000000 -0.0626585802          -0.0634011366
## Expenditure              -0.0626585802  1.0000000000           0.9999557442
## Expenditure.Per.Capita -0.0634011366  0.9999557442           1.0000000000
## Corruption.Index          0.0369408828 -0.0402495916          -0.0401713634
## Exports                   0.0021499215 -0.0129836865          -0.0165026525
## Exports...GDP             0.0029957699 -0.0022549981          -0.0014015194
## Imports                   0.0048630560 -0.0140728548          -0.0182350225
## Imports...GDP             0.0153237733 -0.0396139582          -0.0386766969
## Population                0.0036510335 -0.0064213023          -0.0083440691
## Fertility.Rate            0.0009680047  0.0048172341           0.0051472425
## Crude.death.rate          0.0216248576 -0.0422134871          -0.0423386775
## HDI                      -0.0062676791 -0.0108026336          -0.0110987015
## Life.expectancy          -0.0033794500  0.0006695299           0.0004558625
## CO2.Tons.per.capita      -0.0042694905  0.0028572203           0.0021443102
##                          Corruption.Index      Exports Exports...GDP      Imports
## Annual.GDP                     0.25323253  0.635874084  -0.131377739  0.821331789
## GDP.per.capita                 0.82290513  0.366395569   0.211256695  0.361193202
```

```
## Debt                          0.21515627  0.660986332  -0.133425323  0.786452912
## Debt.Per.Capita               0.63550340  0.482477652   0.123295628  0.490705412
## Deficit                       0.03694088  0.002149922   0.002995770  0.004863056
## Expenditure                  -0.04024959 -0.012983686  -0.002254998 -0.014072855
## Expenditure.Per.Capita       -0.04017136 -0.016502652  -0.001401519 -0.018235022
## Corruption.Index              1.00000000  0.328306564   0.299027298  0.321982893
## Exports                       0.32830656  1.000000000   0.125498692  0.948133191
## Exports...GDP                 0.29902730  0.125498692   1.000000000  0.068021011
## Imports                       0.32198289  0.948133191   0.068021011  1.000000000
## Imports...GDP                 0.17974910  0.013895577   0.884795669 -0.011001650
## Population                   -0.11740148  0.494607545  -0.119815163  0.453347686
## Fertility.Rate               -0.55415821 -0.336255929  -0.257773672 -0.311512767
## Crude.death.rate              0.02400498  0.030397423  -0.067826331  0.029386687
## HDI                           0.79928743  0.392554642   0.268674501  0.382709416
## Life.expectancy               0.64451605  0.338058506   0.211546378  0.320967454
## CO2.Tons.per.capita           0.64007189  0.375023343   0.216333069  0.398192865
##                         Imports...GDP   Population Fertility.Rate
## Annual.GDP                -0.17895417  0.195556688  -0.1924585653
## GDP.per.capita             0.06804813 -0.085113537  -0.4849820314
## Debt                      -0.17758572  0.244810958  -0.1803978351
## Debt.Per.Capita           -0.01069346 -0.046126310  -0.4292734823
## Deficit                    0.01532377  0.003651033   0.0009680047
## Expenditure               -0.03961396 -0.006421302   0.0048172341
## Expenditure.Per.Capita    -0.03867670 -0.008344069   0.0051472425
## Corruption.Index           0.17974910 -0.117401477  -0.5541582053
## Exports                    0.01389558  0.494607545  -0.3362559288
## Exports...GDP              0.88479567 -0.119815163  -0.2577736722
## Imports                   -0.01100165  0.453347686  -0.3115127666
## Imports...GDP              1.00000000 -0.162040565  -0.2129311535
## Population                -0.16204057  1.000000000  -0.0655877212
## Fertility.Rate            -0.21293115 -0.065587721   1.0000000000
## Crude.death.rate          -0.14576248 -0.055289081   0.2419994643
## HDI                        0.14089755 -0.097603505  -0.8402881315
## Life.expectancy            0.16808310 -0.036182005  -0.8601687950
## CO2.Tons.per.capita        0.05485838 -0.020036280  -0.5329793800
##                         Crude.death.rate          HDI Life.expectancy
## Annual.GDP                    0.04633965  0.296429902    0.2232072381
## GDP.per.capita                0.03898162  0.760408695    0.6211771609
## Debt                          0.06860334  0.255577752    0.2100444085
## Debt.Per.Capita               0.10511149  0.647703031    0.5532912787
## Deficit                       0.02162486 -0.006267679   -0.0033794500
## Expenditure                  -0.04221349 -0.010802634    0.0006695299
## Expenditure.Per.Capita       -0.04233868 -0.011098702    0.0004558625
## Corruption.Index              0.02400498  0.799287433    0.6445160467
## Exports                       0.03039742  0.392554642    0.3380585057
## Exports...GDP                -0.06782633  0.268674501    0.2115463782
## Imports                       0.02938669  0.382709416    0.3209674536
## Imports...GDP                -0.14576248  0.140897548    0.1680831035
## Population                   -0.05528908 -0.097603505   -0.0361820052
## Fertility.Rate                0.24199946 -0.840288131   -0.8601687950
## Crude.death.rate              1.00000000 -0.164938738   -0.4337332930
## HDI                          -0.16493874  1.000000000    0.9060353597
## Life.expectancy              -0.43373329  0.906035360    1.0000000000
## CO2.Tons.per.capita           0.02034726  0.691908246    0.5287501612
```

```
##                         CO2.Tons.per.capita
## Annual.GDP                      0.363423071
## GDP.per.capita                  0.696580631
## Debt                            0.304306786
## Debt.Per.Capita                 0.502863415
## Deficit                        -0.004269491
## Expenditure                     0.002857220
## Expenditure.Per.Capita          0.002144310
## Corruption.Index                0.640071888
## Exports                         0.375023343
## Exports...GDP                   0.216333069
## Imports                         0.398192865
## Imports...GDP                   0.054858382
## Population                     -0.020036280
## Fertility.Rate                 -0.532979380
## Crude.death.rate                0.020347257
## HDI                             0.691908246
## Life.expectancy                 0.528750161
## CO2.Tons.per.capita             1.000000000
```

- Choosing: Annual.GDP, Exports, Imports, Corruption.Index, HDI, CO2.Tons.per.capita

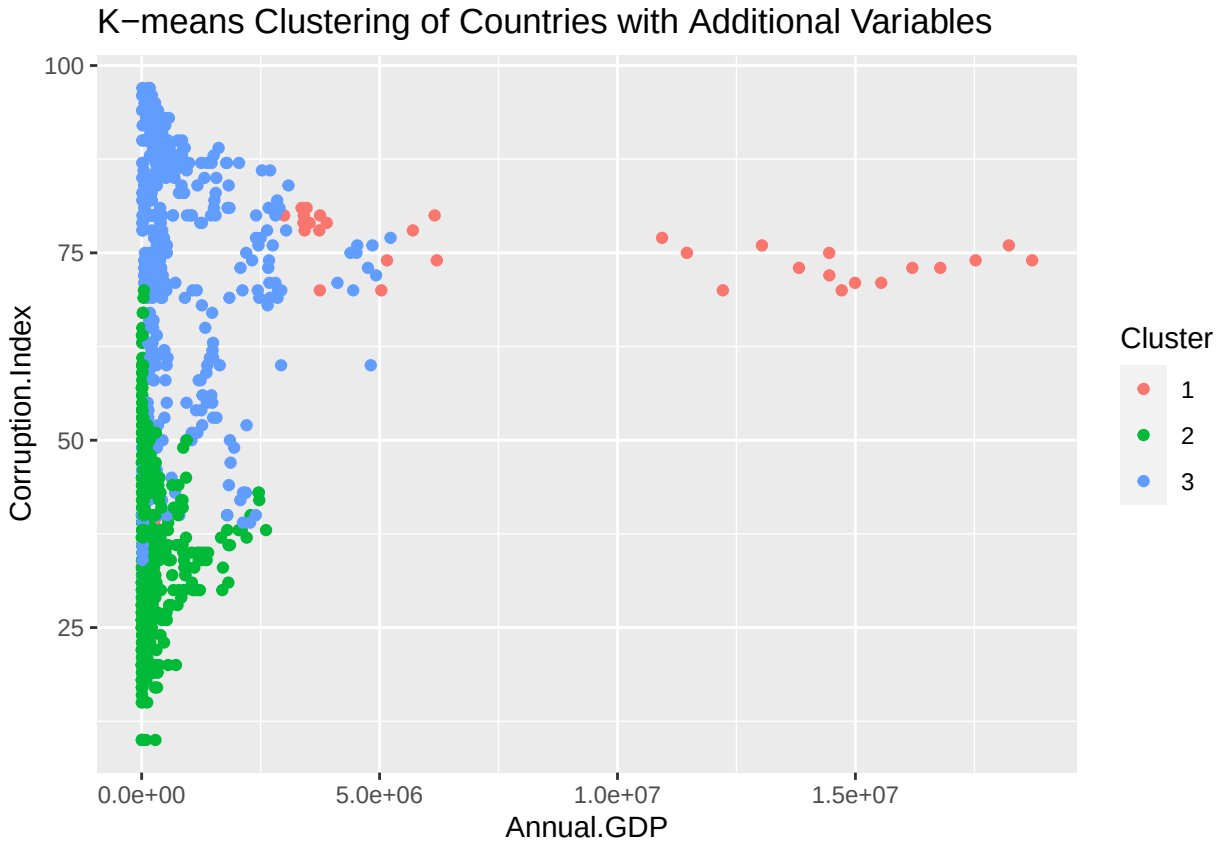
```r
set.seed(123)
economic_data <- country[, c('Annual.GDP', 'Exports', 'Imports', 'Corruption.Index', 'HDI', 'CO2.Tons.pe

# Standardize the data
scaled_economic_data <- scale(economic_data)

# Elbow Method
wss <- numeric(10)  # Adjust the number of clusters based on your analysis

for (i in 1:10) {
  kmeans_model <- kmeans(scaled_economic_data, centers = i, nstart = 25)
  wss[i] <- sum(kmeans_model$withinss)
}

# Plot the elbow
plot(1:10, wss, type = "b", main = "Elbow Method",
     xlab = "Number of Clusters (k)", ylab = "Within-cluster Sum of Squares")
```




- Give that after the 3rd cluster the slope of the graph decreases, we will use 3 clusters

**Identifying Economic Patterns**

```
kmeans_model <- kmeans(scaled_economic_data, centers = 3)
country$Cluster <- as.factor(kmeans_model$cluster)

# Visualize clusters with additional variables
ggplot(country, aes(x = Annual.GDP, y = Corruption.Index, color = Cluster)) +
  geom_point() +
  labs(title = "K-means Clustering of Countries with Additional Variables")
```




```
# Display variable centers for interpretation
kmeans_model$centers
```

```
##   Annual.GDP    Exports    Imports Corruption.Index        HDI
## 1  3.6163547  4.2370947  4.3129014        0.7481841  0.8927747
## 2 -0.2398659 -0.3763763 -0.3545606       -0.6533913 -0.6158410
## 3  0.0763697  0.2559345  0.2106671        1.0699498  0.9906976
##   CO2.Tons.per.capita
## 1           1.3422683
## 2          -0.5886702
## 3           0.9006967
```

- Cluster 1: Higher values in Annual.GDP, Exports, Imports, Corruption.Index, HDI, and CO2.Tons.per.capita. This cluster may represent countries with high economic development, exports, and imports, as well as higher corruption, human development, and CO2 emissions per capita.
- Cluster 2: Lower values in Annual.GDP, Exports, Imports, Corruption.Index, HDI, and CO2.Tons.per.capita. This cluster may represent countries with lower economic development, exports, imports, lower corruption, human development, and lower CO2 emissions per capita.
- Cluster 3: Moderate values in Annual.GDP, Exports, Imports, Corruption.Index, HDI, and CO2.Tons.per.capita. This cluster may represent countries with moderate economic development, exports, imports, corruption, human development, and CO2 emissions per capita.

## Hierarchical Clustering

```
library(vegan)
```

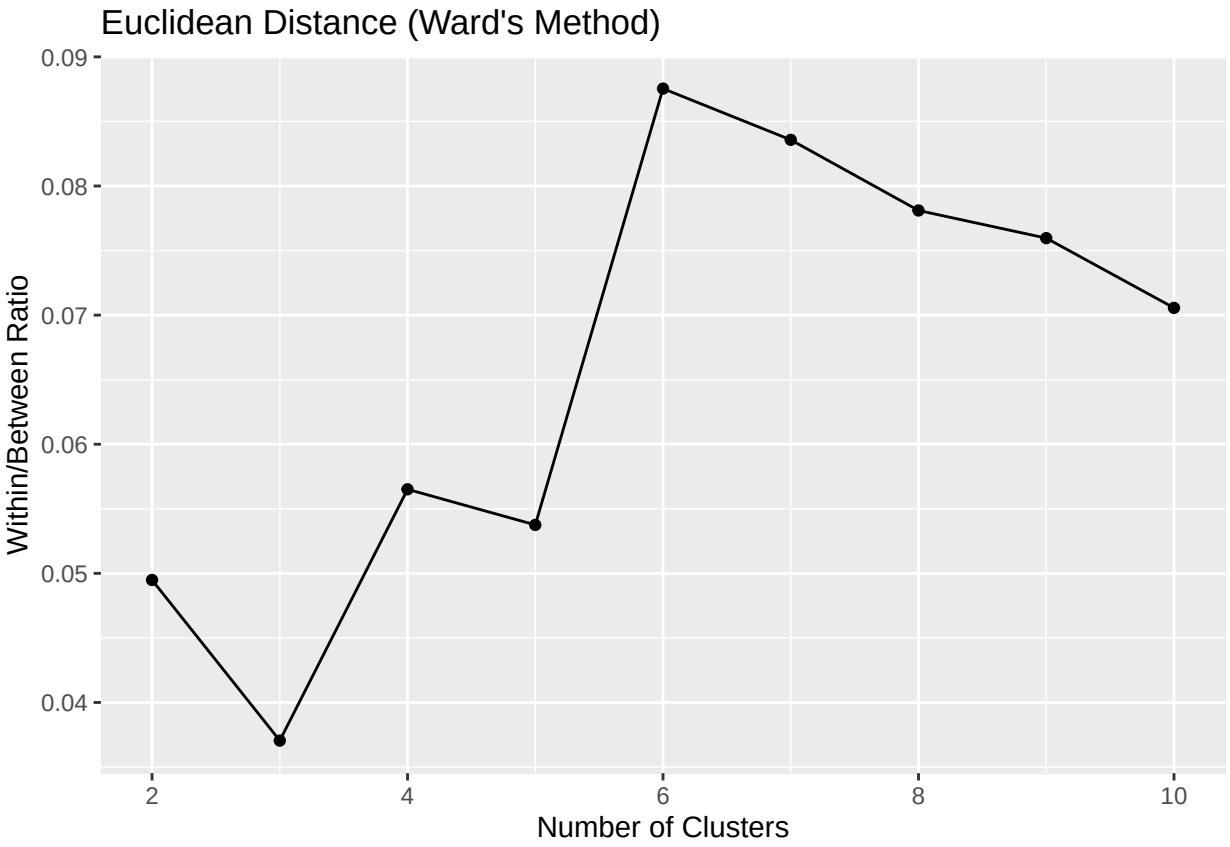
```
## Loading required package: permute
```

```
## This is vegan 2.6-4
```

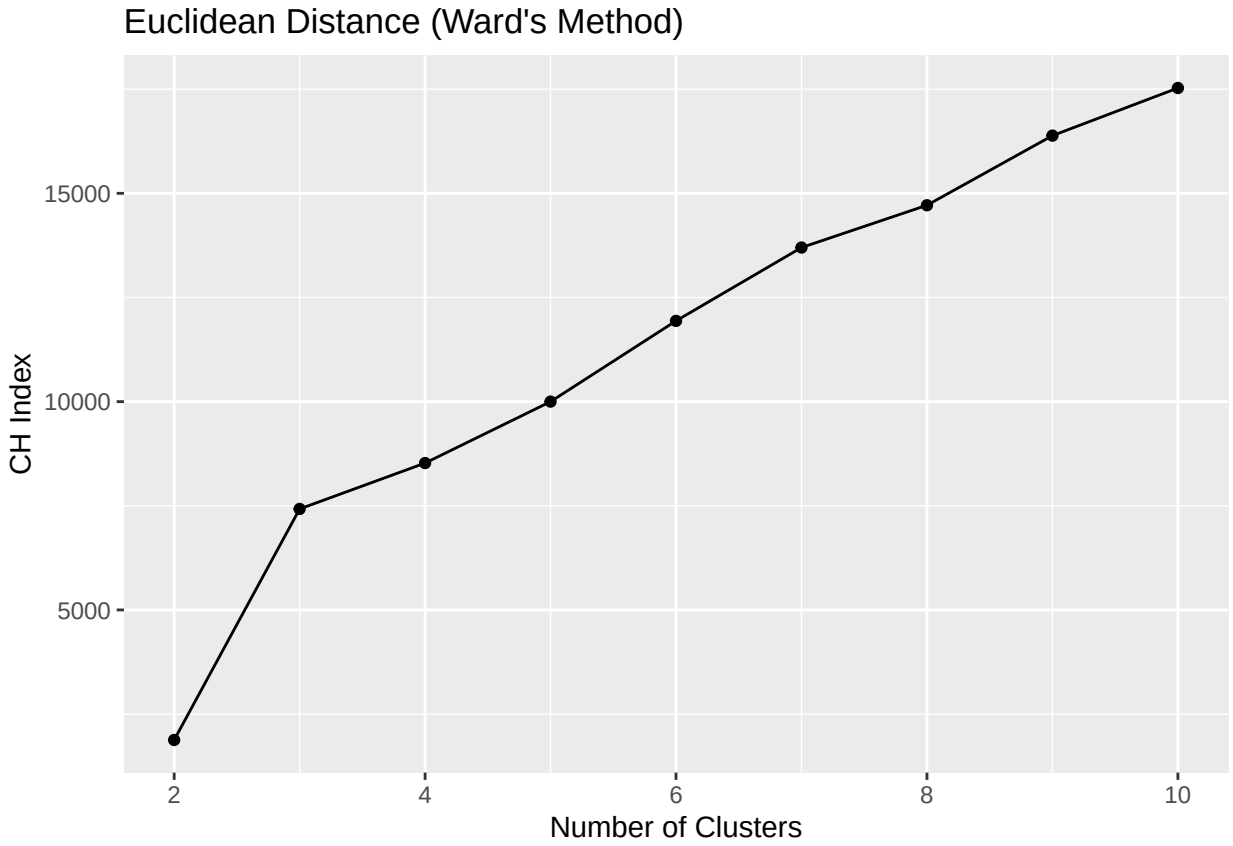
```
##
## Attaching package: 'vegan'
```

```
## The following object is masked from 'package:caret':
##
##     tolerance
```

```
library(fpc)
country.dist <- vegdist(country[,5:22], method = "euclidean")
country.clust <- hclust(country.dist, method = "ward.D2")

# Plot within/between ratios against number of clusters
country.ratio <- sapply(2:10, function(x) cluster.stats(country.dist, clustering = cutree(country.clust
ggplot(data.frame(x = 2:10, y = country.ratio), aes(x, y)) +
  geom_point() +
  geom_line() +
  labs(x = "Number of Clusters", y = "Within/Between Ratio", title = "Euclidean Distance (Ward's Method
```




```
# Plot Calinski-Harabasz index against number of clusters
country.ch <- sapply(2:10, function(x) cluster.stats(country.dist, clustering = cutree(country.clust, x
ggplot(data.frame(x = 2:10, y = country.ch), aes(x, y)) +
  geom_point() +
  geom_line() +
  labs(x = "Number of Clusters", y = "CH Index", title = "Euclidean Distance (Ward's Method)")
```




```
country.clust.cls <- cutree(country.clust, 10)  # 3-cluster model
```

```
library(dendextend)
```

```
## Registered S3 method overwritten by 'dendextend':
##   method     from
##   rev.hclust vegan
```

```
##
## ---------------------
## Welcome to dendextend version 1.17.1
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## You may ask questions at stackoverflow, use the r and dendextend tags:
##   https://stackoverflow.com/questions/tagged/dendextend
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------
```

```
##
## Attaching package: 'dendextend'
```

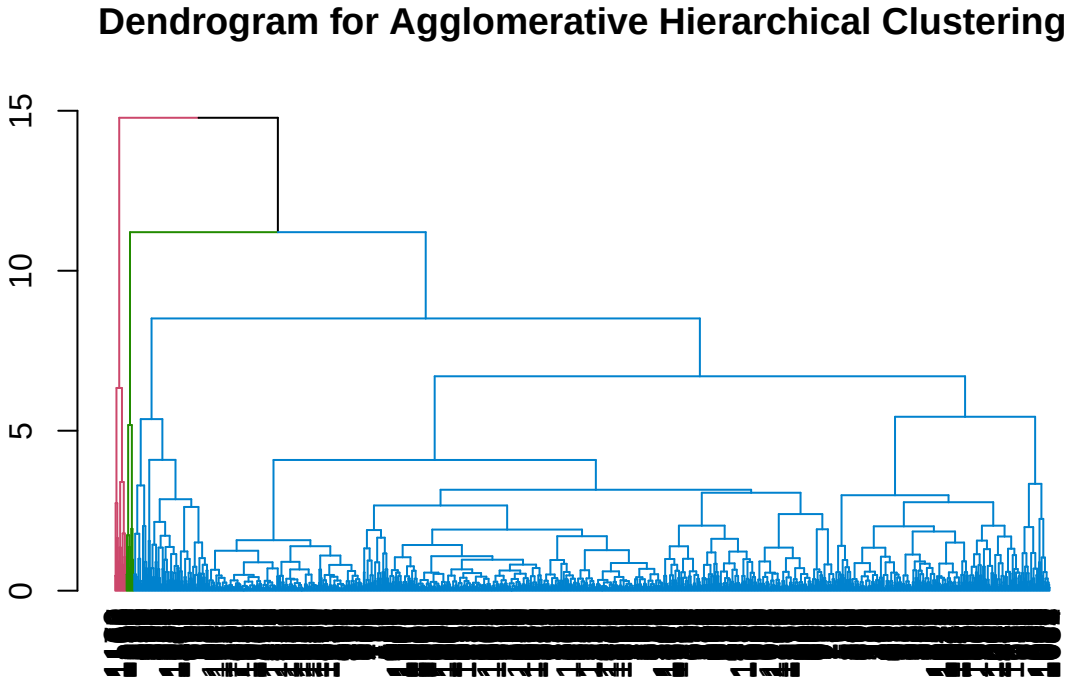
```
## The following object is masked from 'package:permute':
##
##     shuffle
```

```
## The following object is masked from 'package:stats':
##
##     cutree
```

```
library(factoextra)
```

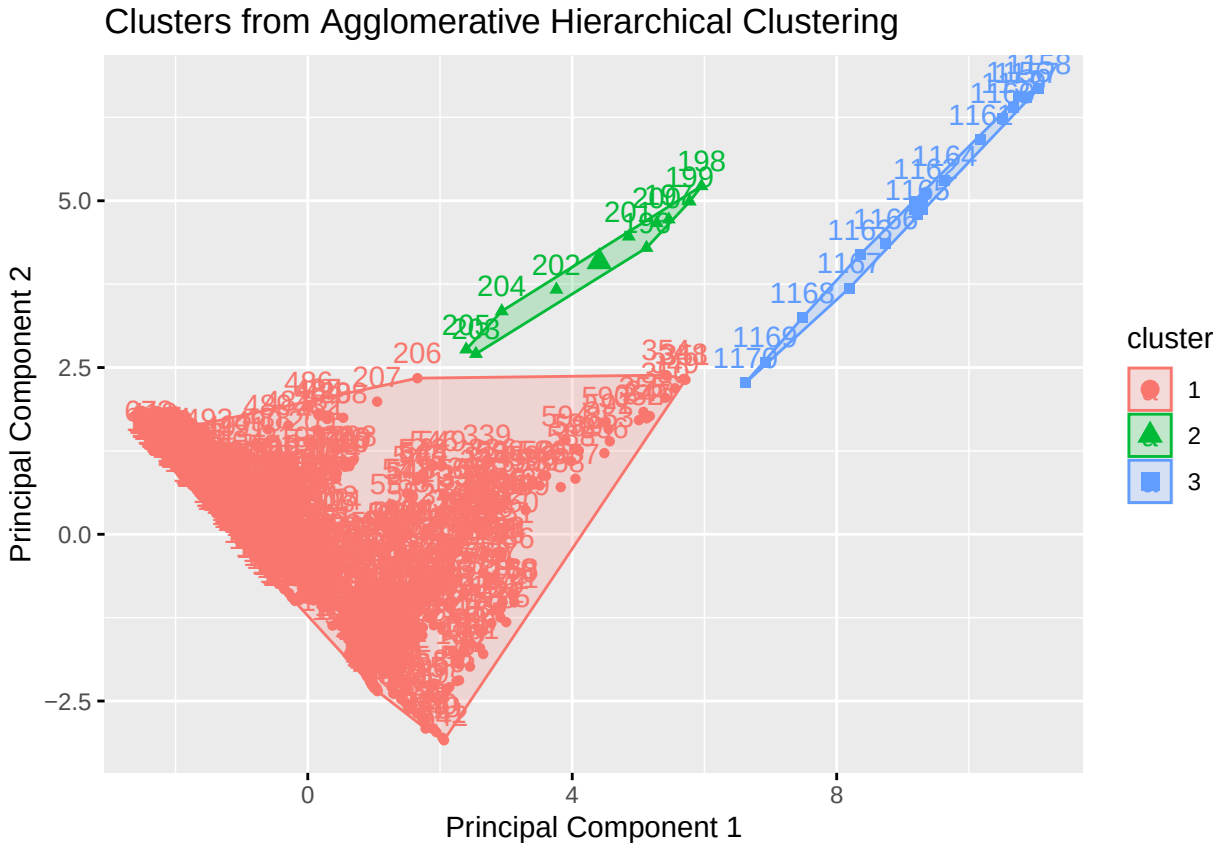
```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(dplyr)
library(tidyr)

economic_data <- country[, c('Annual.GDP', 'Exports', 'Imports', 'Corruption.Index', 'HDI', 'CO2.Tons.p

# Standardize the data
scaled_data <- scale(economic_data)

# Perform hierarchical clustering
hierarchical_clustering <- hclust(dist(scaled_data), method = "complete")

# Cut the dendrogram to get clusters
num_clusters <- 3
cluster_labels <- cutree(hierarchical_clustering, k = num_clusters)

# Add cluster labels to the original data
data_with_clusters <- cbind(scaled_data, Cluster = as.factor(cluster_labels))

# Visualize the dendrogram
dend <- as.dendrogram(hierarchical_clustering)
dend %>%
  set("branches_k_color", k = num_clusters) %>%
  plot(main = "Dendrogram for Agglomerative Hierarchical Clustering")
```




```
# Visualize the clusters in a scatter plot
fviz_cluster(list(data = scaled_data, cluster = cluster_labels)) +
```

```
  labs(title = "Clusters from Agglomerative Hierarchical Clustering",
       x = "Principal Component 1",
       y = "Principal Component 2")
```




```
# Display the first few rows of the data with cluster assignments
data_with_clusters_df <- as.data.frame(data_with_clusters)

head(data_with_clusters_df[data_with_clusters_df$Cluster == 1, ])
```

```
##   Annual.GDP    Exports    Imports Corruption.Index       HDI
## 1 -0.3318962 -0.5119291 -0.4867138       -0.4083919 0.3383859
## 2 -0.3321586 -0.5120790 -0.4877267       -0.5370166 0.2974065
## 3 -0.3311285 -0.5102619 -0.4847322       -0.6656412 0.2769168
## 4 -0.3313848 -0.5105811 -0.4857158       -0.7513910 0.2632569
## 5 -0.3316399 -0.5119011 -0.4858480       -0.6656412 0.2359373
## 6 -0.3313210 -0.5119709 -0.4842081       -0.7513910 0.1334886
##   CO2.Tons.per.capita Cluster
## 1          -0.6851679       1
## 2          -0.6832661       1
## 3          -0.6604448       1
## 4          -0.6851679       1
## 5          -0.6927750       1
## 6          -0.6661501       1
```

```
head(data_with_clusters_df[data_with_clusters_df$Cluster == 2, ])
```

```
##     Annual.GDP  Exports  Imports Corruption.Index         HDI
## 196 -0.1996518 6.914017 4.666835       -0.3655171  0.10616901
```

```
## 197 -0.2031791 7.537086 4.965119       -0.4941417  0.07201946
## 198 -0.1939877 7.780965 5.875412       -0.5370166  0.03786991
## 199 -0.1840774 7.308664 5.845327       -0.3655171 -0.02359928
## 200 -0.1903589 6.740678 5.417029       -0.4083919 -0.07140865
## 201 -0.1986378 6.207978 5.173167       -0.5370166 -0.12604793
##     CO2.Tons.per.capita Cluster
## 196           0.4958372       2
## 197           0.4844266       2
## 198           0.4939355       2
## 199           0.4844266       2
## 200           0.4292750       2
## 201           0.4045518       2
```

```
head(data_with_clusters_df[data_with_clusters_df$Cluster == 3, ])
```

```
##      Annual.GDP  Exports  Imports Corruption.Index      HDI CO2.Tons.per.capita
## 1156  10.042783 4.622737 6.822336        1.0922288 1.294573            2.057198
## 1157   9.770868 4.805443 7.034385        1.1779786 1.280914            2.116153
## 1158   9.383963 5.223429 7.350912        1.0922288 1.267254            2.235966
## 1159   8.972151 5.078364 7.079168        1.0493540 1.253594            2.215046
## 1160   8.646042 4.958276 7.103463        1.0493540 1.267254            2.194126
## 1161   8.283044 4.734346 6.873992        0.9636042 1.260424            2.327251
##      Cluster
## 1156       3
## 1157       3
## 1158       3
## 1159       3
## 1160       3
## 1161       3
```

```
# Calculate mean values for each cluster
cluster_means <- aggregate(data_with_clusters[, 1:6], by = list(Cluster = data_with_clusters_df$Cluster

# Print the results
print(cluster_means)
```

```
##   Cluster  Annual.GDP    Exports    Imports Corruption.Index         HDI
## 1       1 -0.09671525 -0.0963092 -0.1100177     -0.008926156 -0.01399206
## 2       2 -0.21111621  5.9656752  4.4553325     -0.519866631 -0.11648606
## 3       3  7.91020292  3.7597219  5.8678666      1.063645594  1.20168657
##   CO2.Tons.per.capita
## 1         -0.03391492
## 2          0.34806897
## 3          2.49245229
```

Cluster 1: - Annual.GDP: Slightly below the overall mean, indicating a lower GDP on average. - Exports: Similar to Annual.GDP, slightly below the overall mean. - Imports: Slightly below the overall mean, suggesting lower imports. - Corruption.Index: Close to the overall mean, indicating average corruption levels. - HDI (Human Development Index): Close to the overall mean, suggesting average human development. - CO2.Tons.per.capita: Slightly below the overall mean, indicating a relatively lower carbon footprint.

Cluster 2: - Annual.GDP: Considerably below the overall mean, indicating lower GDP. - Exports: Much higher than the overall mean, suggesting a high level of exports. - Imports: Also higher than the overall mean, indicating a high level of imports. - Corruption.Index: Significantly below the overall mean, suggesting lower corruption. - HDI: Below the overall mean, indicating lower human development. - CO2.Tons.per.capita: Above the overall mean, indicating a higher carbon footprint.

Cluster 3: - Annual.GDP: Significantly above the overall mean, indicating higher GDP. - Exports: Above the overall mean, suggesting a moderate level of exports. - Imports: Slightly above the overall mean, indicating a moderate level of imports. - Corruption.Index: Above the overall mean, suggesting a moderate level of corruption. - HDI: Above the overall mean, indicating higher human development. - CO2.Tons.per.capita: Above the overall mean, suggesting a higher carbon footprint.

## Classification - Logistic Regression

- Can we classify countries into status based on the available features.
- In this case we will use logistic regression, we will use all variables in this case to classify continets

```
# Rereading data just to ensure data is clean
country <- read.csv('sample-data.csv')

names(country)
```

```
##  [1] "X.1"                   "X"                     "Country"
##  [4] "Date"                  "Annual.GDP"            "GDP.per.capita"
##  [7] "Debt"                  "Debt.Per.Capita"       "Deficit...M..."
## [10] "Expenditure..M..."     "Expenditure.Per.Capita" "Corruption.Index"
## [13] "Exports"               "Exports...GDP"         "Imports"
## [16] "Imports...GDP"         "Population"            "Fertility.Rate"
## [19] "Crude.death.rate"      "HDI"                   "Life.expectancy"
## [22] "CO2.Tons.per.capita"   "Continent"             "status"
```

```
names(country[,9:10])
```

```
## [1] "Deficit...M..."    "Expenditure..M..."
```

```
names(country)[names(country) == "Expenditure..M..."] <- "Expenditure"
names(country)[names(country) == "Deficit...M..."] <- "Deficit"
names(country[,9:10])
```

```
## [1] "Deficit"     "Expenditure"
```

- We will use stepwise selection
- Based on correlations previously we saw that annual.GDP and DEBT were highly correlated so I decided to remove those to avoid multicollinearity
- I believe imports, expenditure and exports are better economic variables that can help explain the previous 2 variables

```
library(tidyverse)

country$status <- as.factor(country$status)
country$status_code <- as.numeric(country$status == "Developed")

# Split the data into training and testing sets
set.seed(123)
splitIndex <- createDataPartition(country$status_code, p = 0.8, list = FALSE)
train_data <- country[splitIndex, ]
test_data <- country[-splitIndex, ]

# Perform logistic regression with stepwise variable selection
initial_model <- glm(status_code ~ Expenditure+CO2.Tons.per.capita++Expenditure+HDI+Corruption.Index+
                     Fertility.Rate+Crude.death.rate+Life.expectancy, family = 'binomial', data = train

step_model <- step(initial_model, direction = "both", trace = FALSE)
```

```r
# Assess model performance (you may want to split your data into training and testing sets for a more r
predicted_probs <- predict(step_model, type = "response")
predicted_classes <- ifelse(predicted_probs > 0.5, 1, 0)


observed_classes <- train_data$status_code

# Assess model performance
conf_matrix <- table(observed_classes, predicted_classes)
accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
print(paste("Accuracy:", accuracy))
```

```
## [1] "Accuracy: 0.960365853658537"
```

```r
coefficients <- coef(step_model)
print(coefficients)
```

```
##         (Intercept) CO2.Tons.per.capita                 HDI    Corruption.Index
##        -69.02827288         -0.11069457         65.33717546           0.03328838
##    Crude.death.rate
##       1501.31408730
```

```r
# Precision
precision <- conf_matrix[2, 2] / sum(conf_matrix[, 2])
print(paste("Precision:", precision))
```

```
## [1] "Precision: 0.906752411575563"
```

- The overall accuracy of 96% suggests that the model is performing well in terms of correct classification.
- The high precision indicates that when the model predicts a country as "Developed," it is likely to be correct about 90% of the time.

**Health and Human Development:**

**Decisions Tree**

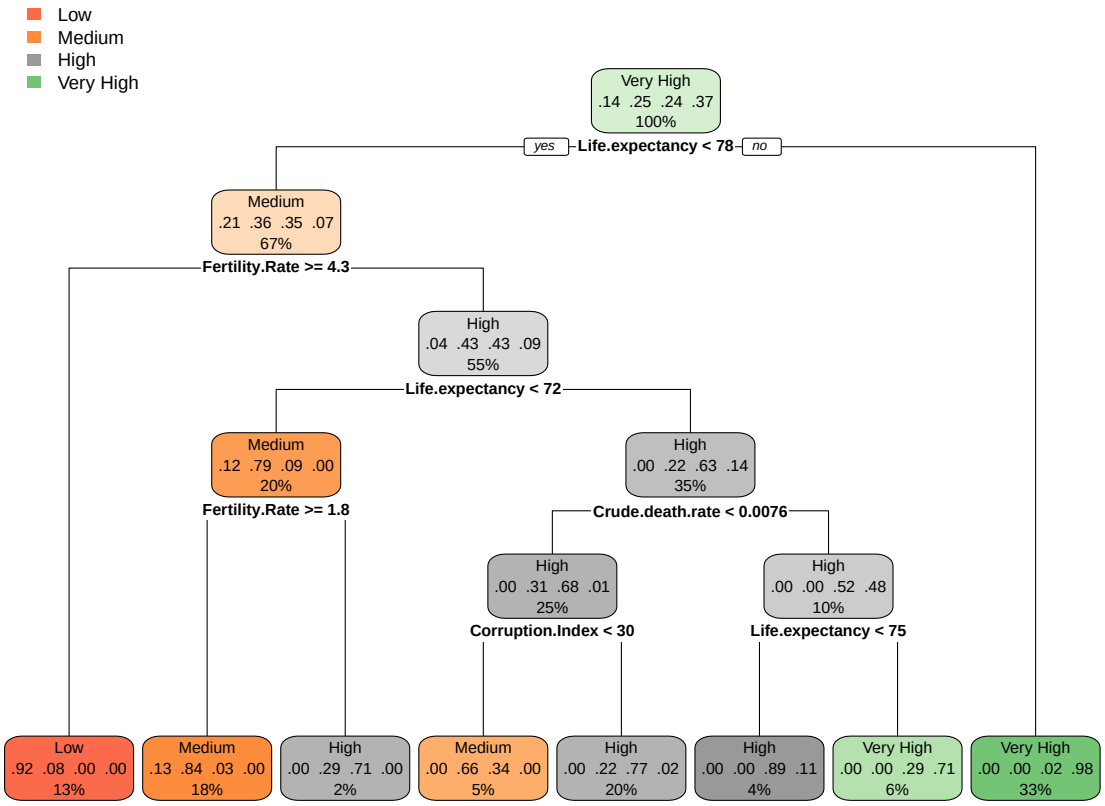
- Classify countries into high, medium, or low human development categories based on HDI (human development index), life expectancy, fertility rate, and other relevant features.
- According to United Nations Development Program:
- = .8 VERY HIGH DEVELOPMENT
- .7-.7999 HIGH DEVELOPMENT
- .55-.699 MEDIUM DEVELOPMENT
- <.55 LOW DEVELOPMENT

```r
# Load necessary libraries
library(rpart)
```

```
##
## Attaching package: 'rpart'
```

```
## The following object is masked from 'package:dendextend':
##
##     prune
```

```r
library(rpart.plot)
library(caret)
library(dplyr)

df <- read.csv("sample-data.csv")

# Create a new categorical variable for Human Development Level based on HDI
df$Human_Development_Level <- cut(df$HDI,
                                  breaks = c(-Inf, 0.549, 0.699, 0.799, Inf),
                                  labels = c("Low", "Medium", "High", "Very High"))

features <- c("Life.expectancy", "Fertility.Rate", "Crude.death.rate", "Corruption.Index")
target <- "Human_Development_Level"

# Create a new dataframe with selected features
df_selected <- df[, c(features, target)]

# Convert categorical variables to factors
df_selected$Human_Development_Level <- as.factor(df_selected$Human_Development_Level)

set.seed(150)
train_index <- createDataPartition(df_selected$Human_Development_Level, p = 0.8, list = FALSE)
train_data <- df_selected[train_index, ]
test_data <- df_selected[-train_index, ]

# Train a decision tree model
tree_model <- rpart(Human_Development_Level ~ Life.expectancy + Fertility.Rate + Crude.death.rate + Cor

# Visualize the decision tree
rpart.plot(tree_model)
```




```
# Make predictions on the test set
predictions <- predict(tree_model, newdata = test_data, type = "class")

# Confusion matrix
conf_matrix <- confusionMatrix(predictions, test_data$Human_Development_Level)
print(conf_matrix)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Low Medium High Very High
##   Low        28      2    0         0
##   Medium      6     52    6         0
##   High        0      6   47         4
##   Very High   0      0    7        86
##
## Overall Statistics
##
##                Accuracy : 0.873
##                  95% CI : (0.8245, 0.912)
##     No Information Rate : 0.3689
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8236
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
```

```
##                      Class: Low Class: Medium Class: High Class: Very High
## Sensitivity              0.8235        0.8667      0.7833           0.9556
## Specificity              0.9905        0.9348      0.9457           0.9545
## Pos Pred Value           0.9333        0.8125      0.8246           0.9247
## Neg Pred Value           0.9720        0.9556      0.9305           0.9735
## Prevalence               0.1393        0.2459      0.2459           0.3689
## Detection Rate           0.1148        0.2131      0.1926           0.3525
## Detection Prevalence     0.1230        0.2623      0.2336           0.3811
## Balanced Accuracy        0.9070        0.9007      0.8645           0.9551
```

- The high accuracy, sensitivity, specificity, and precision values suggest that the decision tree model performs exceptionally well across all classes.
- The model predicted 28 instances correctly as "Low," 52 instances correctly as "Medium," 47 instances correctly as "High," and 86 instances correctly as "Very High."
- The overall accuracy of the model is approximately 87.3%, which means the model correctly predicted the class for about 87.3% of the instances.

```r
# Load necessary libraries
library(MASS)  # For LDA
```

**Using LDA and QDA to see if we can predict Human Development Levels**

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```r
features <- c("Annual.GDP", "Expenditure..M...", "Corruption.Index", "Exports",
              "Imports", "Population", "Fertility.Rate",
              "Crude.death.rate", "Life.expectancy", "CO2.Tons.per.capita")

target <- "Human_Development_Level"


df_selected <- df[, c(features, target)]

# Convert categorical variables to factors
df_selected$Human_Development_Level <- as.factor(df_selected$Human_Development_Level)


set.seed(123)
train_index <- createDataPartition(df_selected$Human_Development_Level, p = 0.8, list = FALSE)
train_data <- df_selected[train_index, ]
test_data <- df_selected[-train_index, ]

lda_model <- lda(Human_Development_Level ~ ., data = train_data)

# Make predictions using LDA on the test set
lda_predictions <- predict(lda_model, newdata = test_data)

# Display LDA confusion matrix and assess performance
lda_conf_matrix <- table(lda_predictions$class, test_data$Human_Development_Level)
print("LDA Confusion Matrix:")
```

```
## [1] "LDA Confusion Matrix:"
```

```
print(lda_conf_matrix)
```

```
##
##             Low Medium High Very High
##   Low        33      1    0         0
##   Medium      1     50    6         0
##   High        0      9   45         5
##   Very High   0      0    9        85
```

```
lda_accuracy <- sum(diag(lda_conf_matrix)) / sum(lda_conf_matrix)
print(paste("LDA Accuracy:", lda_accuracy))
```

```
## [1] "LDA Accuracy: 0.872950819672131"
```

```
# Train the QDA model
qda_model <- qda(Human_Development_Level ~ ., data = train_data)

# Make predictions using QDA on the test set
qda_predictions <- predict(qda_model, newdata = test_data)

# Display QDA confusion matrix and assess performance
qda_conf_matrix <- table(qda_predictions$class, test_data$Human_Development_Level)
print("QDA Confusion Matrix:")
```

```
## [1] "QDA Confusion Matrix:"
```

```
print(qda_conf_matrix)
```

```
##
##             Low Medium High Very High
##   Low        33     18    0         0
##   Medium      1     15    0         0
##   High        0     27   58         6
##   Very High   0      0    2        84
```

```
qda_accuracy <- sum(diag(qda_conf_matrix)) / sum(qda_conf_matrix)
print(paste("QDA Accuracy:", qda_accuracy))
```

```
## [1] "QDA Accuracy: 0.778688524590164"
```

- Accuracy: 87.29% Interpretation: LDA has a higher accuracy compared to QDA. It performs well in correctly classifying instances, especially in the "Low" and "Very High" categories.
- Accuracy: 77.86% Interpretation: QDA has a lower accuracy compared to LDA. It struggles in particular with the "Medium" category, misclassifying more instances as "High" and "Low."
- LDA appears to be more accurate in this context, providing a better overall classification performance.

```
precision_lda <- diag(lda_conf_matrix) / rowSums(lda_conf_matrix)
print("Precision - LDA:")
```

```
## [1] "Precision - LDA:"
```

```
print(precision_lda)
```

```
##       Low    Medium      High Very High
## 0.9705882 0.8771930 0.7627119 0.9042553
```

```r
precision_qda <- diag(qda_conf_matrix) / rowSums(qda_conf_matrix)
print("Precision - QDA:")
```

```
## [1] "Precision - QDA:"
```

```r
print(precision_qda)
```

```
##       Low    Medium      High Very High
## 0.6470588 0.9375000 0.6373626 0.9767442
```

- LDA:
- Out of the instances predicted as “Low,” 97.05% actually belong to the “Low” class.
- Out of the instances predicted as “Medium,” 87.79% actually belong to the “Medium” class.
- Out of the instances predicted as “High,” 76.27% actually belong to the “High” class.
- Out of the instances predicted as “Very High,” 90.4% actually belong to the “Very High” class.
- QDA:
- Out of the instances predicted as “Low,” 64.70% actually belong to the “Low” class.
- Out of the instances predicted as “Medium,” 93.75% actually belong to the “Medium” class.
- Out of the instances predicted as “High,” 63.73% actually belong to the “High” class.
- Out of the instances predicted as “Very High,” 97.67% actually belong to the “Very High” class.